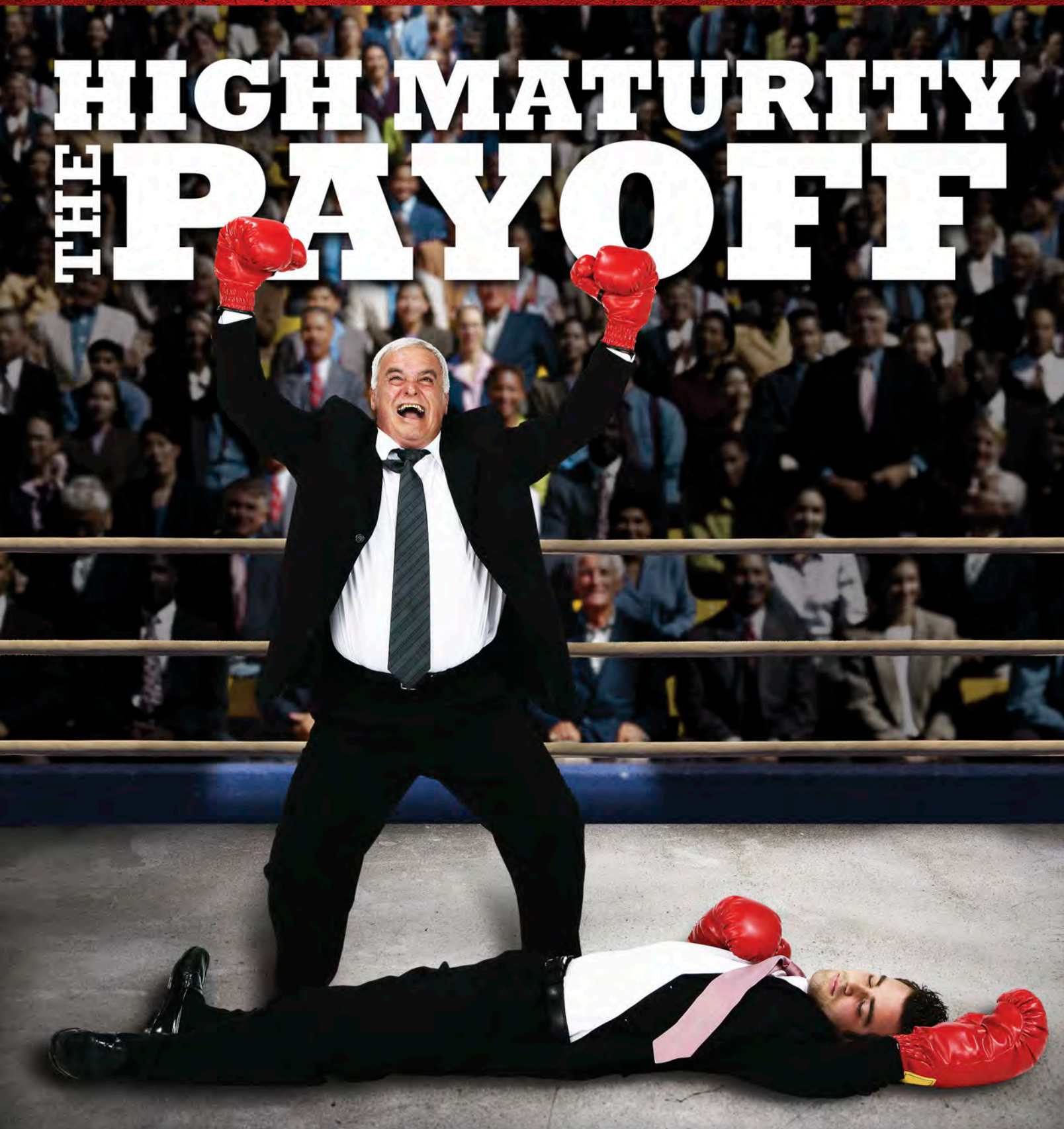
CrossTalk

Jan / Feb 2012 The Journal of Defense Software Engineering Vol. 25 No. 1

THE HIGH MATURITY PAYOFF



Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
FEB 2012		**00-00-2012 to 00-00-2012**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
CrossTalk: The Journal of Defense Software Engineering	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **517 SMXS MXDEA,6022 Fir Ave,Hill AFB,UT,84056**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

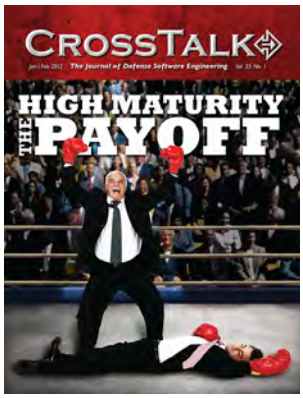
14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**	**Same as Report (SAR)**	**40**	

CONTENTS

Departments



Cover Design by Kent Bingham

High Maturity - The Payoff

CrossTalk

NAVAIR Jeff Schwalb
DHS Joe Jarzombek
309 SMXG Karl Rogers

Publisher Justin T. Hill
Advisor Kasey Thompson
Article Coordinator Lynne Wade
Managing Director Brent Baxter
Managing Editor Brandon Ellis
Associate Editor Colin Kelly
Art Director Kevin Kiernan

Phone 801-775-5555
E-mail stsc.customerservice@hill.af.mil
CrossTalk Online www.crosstalkonline.org

CrossTalk, The Journal of Defense Software Engineering is co-sponsored by the U.S. Navy (USN); U.S. Air Force (USAF); and the U.S. Department of Homeland Defense (DHS). USN co-sponsor: Naval Air Systems Command. USAF co-sponsor: Ogden-ALC 309 SMXG. DHS co-sponsor: National Cyber Security Division in the National Protection and Program Directorate.

The USAF Software Technology Support Center (STSC) is the publisher of **CrossTalk** providing both editorial oversight and technical review of the journal. **CrossTalk's** mission is to encourage the engineering development of software to improve the reliability, sustainability, and responsiveness of our warfighting capability.

Subscriptions: Visit <www.crosstalkonline.org/subscribe> to receive an e-mail notification when each new issue is published online or to subscribe to an RSS notification feed.

Article Submissions: We welcome articles of interest to the defense software community. Articles must be approved by the **CrossTalk** editorial board prior to publication. Please follow the Author Guidelines, available at <www.crosstalkonline.org/submission-guidelines>. **CrossTalk** does not pay for submissions. Published articles remain the property of the authors and may be submitted to other publications. Security agency releases, clearances, and public affairs office approvals are the sole responsibility of the authors and their organizations.

Reprints: Permission to reprint or post articles must be requested from the author or the copyright holder and coordinated with **CrossTalk.**

Trademarks and Endorsements: **CrossTalk** is an authorized publication for members of the DoD. Contents of **CrossTalk** are not necessarily the official views of, or endorsed by, the U.S. government, the DoD, the co-sponsors, or the STSC. All product names referenced in this issue are trademarks of their companies.

CrossTalk Online Services:
For questions or concerns about crosstalkonline.org web content or functionality contact the **CrossTalk** webmaster at 801-417-3000 or webmaster@luminpublishing.com.

Back Issues Available: Please phone or e-mail us to see if back issues are available free of charge.

CrossTalk is published six times a year by the U.S. Air Force STSC in concert with Lumin Publishing <luminpublishing.com>.
ISSN 2160-1577 (print); ISSN 2160-1593 (online)

CrossTalk would like to thank 309 SMXG for sponsoring this issue.

High Maturity: the Payoff

The articles in this issue of CrossTalk discuss the payoff of high maturity software processes. For years, I along with many of my colleagues have had in-depth discussions about the merits and potential drawbacks of high maturity software processes. In fact I can remember having similar debates almost 20 years ago when 309 SMXG first embarked on CMM® process improvement. I think this debate will continue for the foreseeable future. About two and a half years ago, the Air Force Material Command's three Software Maintenance Groups (SMXGs) formed a software enterprise. This enterprise is comprised of the Software Maintenance Groups from the three Air Force Air Logistics Centers at Hill Air Force Base, Warner Robbins Air Force Base and Tinker Air Force Base. The enterprise is comprised of more than 2,100 engineers and computer scientists whose focus is providing high quality software on time, and within cost, for Air Force weapons systems. This enterprise provides a single software perspective for Air Force Material Command leadership and in some cases Air Force leadership. One of the first things the three SMXG directors did after forming the software enterprise was agree to the pursuit of high maturity software processes across the three groups. The enterprise leadership meets about twice a year to share good ideas ranging from management to process improvement. Under my direction, the 309th SMXG at Hill AFB has spent the last few years working toward implementation of high maturity CMMI® Level 5. Even though our course toward high maturity CMMI has been set, there continues to be debate within the organization about the value of high maturity CMMI Level 5.

I am a strong proponent of high maturity process improvement, however, within SMXG there are still some who doubt the validity of the benefits of high maturity CMMI Level 5 software processes. Most of the doubt seems to stem from the financial investment and the perceived lack of flexibility required by high maturity processes. Most do not argue the validity of high maturity process to improve quality, reliability, and the ability to leverage lessons learned within the group. This ongoing debate is what makes this issue of **CrossTalk** so interesting.

Articles in this issue provide a wealth of information from those who have achieved high maturity CMMI Level 5. The authors address many of the issues surrounding the debate over high maturity software processes. I am excited to utilize the information in this issue to improve future discussions of high maturity process improvement not only within the 309 SMXG, but also across the larger software enterprise and industry.

As we continue to learn about high maturity software processes, we will progress toward better software processes and management techniques. I would like to thank all those who took the time to provide articles for this issue of **CrossTalk**.

Karl Rogers
Director
309th Software Maintenance Group

Do Not Get Out of Control: Achieving Real-time Quality and Performance

Craig Hale, Esterline Control Systems - AVISTA
Mike Rowe, Esterline Control Systems – AVISTA

Abstract. When lives are at risk if systems fail it is critical to minimize defects through the best software engineering processes possible. High-maturity processes are valuable for delivering quality, mission-critical software and supporting overall project performance. One standard tool used is Statistical Process Control (SPC). This allows a process to be monitored in real time to detect problems and eliminate their root causes. It also helps discover beneficial process improvements, so related organizational process improvements can be incorporated.

Introduction

Imagine the ability to see and adjust an activity before problems result. In a way, software development teams have this ability using high-maturity processes, such as those prescribed by CMMI® Maturity Level 5, to optimize and maintain performance levels. Certification and maintenance at that level involves organizational performance management [1]. This encourages organizations to make use of process metrics to refine and optimize their processes.

SPC is a technique that facilitates the monitoring of real-time process performance using control charts. Control charts plot key process parameters against historical organizational standards. One parameter of importance to many software organizations in our industry, and that will be utilized in this article to illustrate the technique, is defects per 1,000 lines of source code (KSLOC).

Run rules help identify non-random variations in process control charts. When non-random variation occurs, a process is considered to be "out of control." An out of control process triggers intervention to determine if anything outside the expected process performance has occurred using root cause analysis.

For example, if project performance to the defects per KSLOC baseline is determined to be out of control, then it is an important event and the project team should understand what has caused this to occur. Since SPC helps monitor process in real time, the root cause of this out of control event is probably still present. If an event is having negative consequences, then identifying and removing the root cause before too much damage has occurred should result in less rework time. If an event has a positive effect to the project, then the project team may change the process to encourage the continuation of the event. Let us look at how this works.

Control Chart Basics

Control charts provide a real-time graphical presentation of how a process is performing in relationship to a historical baseline. A historical baseline is produced by collecting and analyzing previous process data. Thus, it represents an organization's known process capability. If a process is not changed, one would expect that an organization's future process performance would fall within normal variation of this historical baseline.

Using the defects per KSLOC example, let us say an organization has historically been averaging 0.5 defects per KSLOC. There may be no concern if a data point jumps up to 1.5 defects per KSLOC. This is probably within normal variation of historical performance. But, if the figure goes above 4.0, or remains constantly above 1.5, then this might be a significant event. How do we determine whether to take action? This is where the additional components of a control chart come into play.

SPC generally makes use of two different charts simultaneously. One monitors actual process values or averages of values, and the other monitors process variability, such as a standard deviation or range.

Some software organizations use XmR control charts. The two sample control charts here illustrate how code defects per requirement review can be charted. The X chart plots and monitors actual individual process values, such as the total defects per KSLOC from each code review (see Figure 1). The mR chart plots and monitors a moving range (see Figure 2). A moving range is the absolute value of the difference in defects per KSLOC of two sequential code reviews. From the mR chart we can derive what to expect for normal variation.

Each control chart has several additional components that are useful in monitoring a process. The primary component of each is the data line, plotted with connected dots. The data points from the data line are labeled as "Included Data" on the X chart and "mR" on the mR chart. The lines represent the performance of a project's process.

The small "X"s on each chart are data points excluded from control chart computations. These points were investigated and were excluded from the included data and future calculations. One reason to exclude a data point would be that a defect was recorded against a software module because the associated requirement conflicted with another requirement. This type of defect is not really a code defect. It is a defect that was injected into the system at the requirements development stage. It is important to understand the capabilities and performance of defect injection in the code development process. Requirements-injected defects are monitored by another set of control charts.

The Center Line (CL) is the target value that we expect our process to perform around. For the X chart, the CL represents the unweighted average of the historical defects per KSLOC per

review for all similar projects within an organization. Thus, we would expect to see any typical code review utilizing this process to produce about the same number of defects per KSLOC.

In Figure 1, the CL is at 0.57. In the mR chart the CL is the unweighted average of the historical moving ranges for an organization. In Figure 2, this CL line is at 1.03. An unweighted average is a simple average without regard to the size or number of KSLOC in a review. A weighted average would take into account the size of each review when calculating this average.

Other components are Control Limits: Upper (UCL) and Lower (LCL). The UCL is generally set at 3.0 standard deviations above the CL, and the LCL is typically set at 3.0 standard deviations below the CL. In Figure 1 the UCL is at 3.3, and in Figure 2 the UCL is at 3.4. Setting the UCL at 3.0 standard deviations represents probability levels of roughly 0.001. A code review with 4.0 defects per KSLOC (see point 16 in Figure 1) exceeds the UCL of the X chart, indicating an unlikely occurrence purely by chance.

Since defects per KSLOC are dealing with small numbers, and obviously defects per KSLOC cannot go below zero defects per KSLOC, the LCL for the X chart is not used. For the mR chart, the LCL is set to 0.0 for obvious reasons. Although it is rare in software SPC, some processes may utilize the actual LCLs for X charts.

Some organizations select tighter control limits by setting them at a level less than 3.0 standard deviations. This will increase the number of events that trigger out of control events to investigate and the number of false alarms.

The shaded bands, or zones, on the X chart can help detect series of statistically unnatural events using run rules [2]. In SPC there are many possible run rules. For example, our organization uses four traditional run rules. They provide the power to discover unnatural variation for our needs. For organizations interested in information about run rules, Nelson Run Rules are a good starting point [3].

The zones are set at successive 1.0 standard deviation intervals from the CL. Zone C is within 1.0 standard deviations; zone B is from 1.0 to 2.0 standard deviations; and zone A is from 2.0 to 3.0 standard deviations.

If we observe two out of three consecutive points in zone A, then a run rule named "2 out of 3 in one A" triggers. The actually probability of such an event is, at most, 1.5 chances out of 1000. In addition to this run rule, we also monitor run rules called "4 points (on the same side of the CL) out of 5 in zone A or B" and "8 consecutive on the same side of the CL."

Finally, a grand mean is plotted on the control charts (see the dashed line on Figure 1). The grand mean is a weighted average of all defects divided by all KSLOC. Remember, the CL uses an unweighted average of defects per KSLOC per review. If a review had 2.0 KSLOC, it is counted with equal weight as a review containing 0.1 KSLOC. Grand means are generally not part of an XmR control chart, but they are useful and give the project team an overall sense of what the total defect rate for the project is running. Our team uses this defect rate to estimate rework effort.

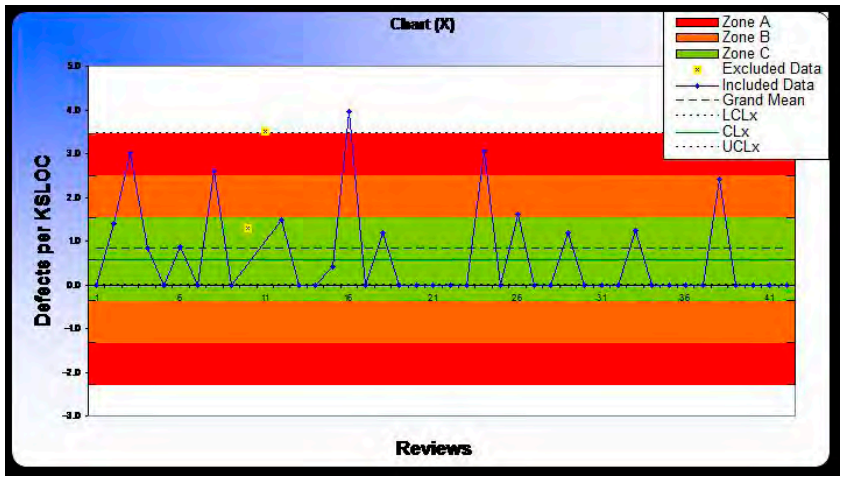


Figure 1: X Control Chart

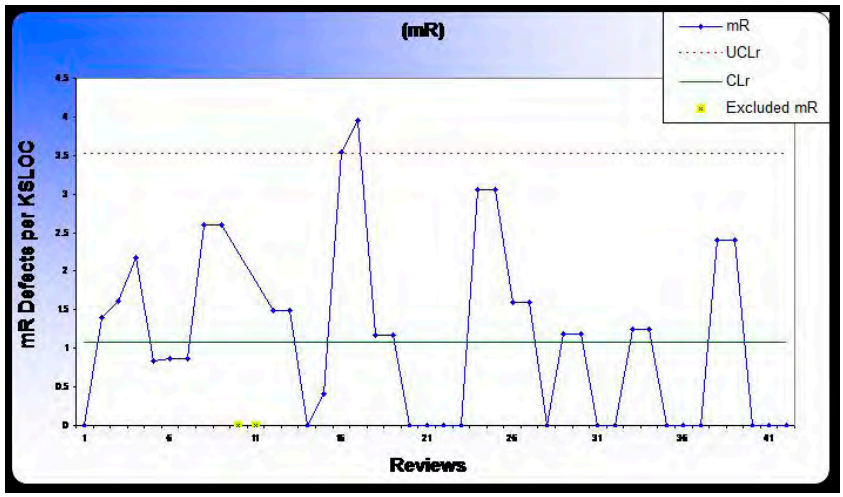


Figure 2: mR Control Chart

The original run rules for SPC were developed in the mid-1950s at a point prior to portable calculators. Using run rules is still standard SPC practice as they provide a quick and understandable way of identifying non-normal events without deep knowledge of statistics.

Control Chart Evolution: A Case Example

This section describes the evolution of the control charts used in our organization to develop life- and mission-critical software in a project-based environment. The evolution of our charts is driven by the data that we collect. With more data, we discover different, and often better, ways to utilize the data.

Our organization started looking at the "what and how" of measuring process performance back in 2004. We did this by defining organization-wide project metrics. Activities addressed include planning, system requirements, software requirements, software design, software coding, etc. These are based on regulatory requirements, such as DO-178B, "Software Considerations in Airborne Systems and Equipment Certification" [4], which regulates avionics software. Our team identified key quality and performance measures that could easily be captured and would provide meaningful information.

> Tracking total defects for a project is not very useful, since a review may have more total defects because they are developing a high number of lines of code. A smaller review may have fewer defects just because of the small size of the review. The performance of the process is understood by evaluating the variance. So, we derive measures to monitor, for instance defects per requirement or defects per KSLOC.

Within the code development activity, it is easy to track lines of source code, number of requirements implemented, code development time, code review time, number of defects, rework time (fixing defects), and rework verification time (time spent ensuring the defects were fixed properly). We track key activities utilizing some existing tools within our organization, with minimal burden on engineers or managers. Today there are COTS tools many organizations can use to do this.

Next came the development of organizational baseline control charts. Before claiming an organization baseline, our team requires two criteria be satisfied. First, there must be enough data points to ensure the baseline is statistically stable—a minimum of 40. Second, there must be a mix of enough different projects to generalize the findings to other projects. This is more of a qualitative decision based on knowledge of the projects.

After generating the first baselines, we noticed very high UCLs, indicating a great deal of variance. With a high UCL, some projects never generated out of control points, whereas other projects consistently triggered run rules. This was the result of developing a mixture of complex systems.

The team looked into the types of complex projects involved, such as avionics and medical systems. Organizational baselines were then split by application area. The resulting medical control charts became more useful at detecting outliers, but the avionics control charts were still not performing to the level desired.

The DO-178B guideline for avionics classifies features based on the criticality of failure from Level A to E. Failure of a Level A feature would result in a catastrophic failure condition for an aircraft. Level A design, development, testing and defects are treated much different than Level E failure, which are associated with defects that will not have an effect on aircraft operational capability or pilot workload [4].

Initially, these levels appeared to be a great way to partition the work. However, the problem partitioning baseline control charts by DO-178B is that a project may have requirements at multiple levels, so it is too difficult to accurately track all measures to these levels.

The team also looked at whether the project dealt predominantly with embedded or non-embedded software. This partitioning could be applied once at the very beginning of a project, simplifying the identification of the project type. Embedded avionics and non-embedded avionics organization control charts had very different CLs and UCLs. The variation from the control limits to the CL was statistically significant for the two control charts. But, we were still not done.

Within the embedded/non-embedded control charts, some reviews had much lower defect rates than others. The major differences were based on what was being reviewed. Higher defect rates were associated with new code that was reviewed for the very first time. Lower defect rates were associated with existing code that had been reworked and were undergoing a subsequent review due to additional code change.

Certain projects differed based on customer and type of work, so customer- and job-specific control chart baselines were created. Some of these differences are based on a customer's process maturity levels. A customer's process maturity level can impact how well they define system and software-related requirements. It also influences how stable the system and software requirements or target hardware remain during a project.

Even with stable organizational baselines the team strives to reexamine the CLs and control limits every six months. This ensures the process improves over time, since as a service-based organization our customer and project mix is changing over time, and with new technologies.

Hypothesis Testing

Evidence is gathered when considering whether to partition an organizational control chart. Some evidence is based on knowledge and experience with our business. Our team also uses statistical tools to identify and study potential differences.

When an organizational control chart is updated, part of the process is to look for differences among customers and projects. When differences are identified, they are statistically analyzed using t-tests or analysis of variance to determine whether the differences are actually statistically significant.

When customer or project data are statistically different the project management team determines why the difference exists. This drives the decision as to whether a more specific organiza-

tional control chart should be created. A new organizational chart is warranted if we expect to be doing similar projects in the future. One-time projects that are unlikely to be repeated are excluded from the organizational control charts. The information is documented just in case projects similar to this do start popping up.

Benefits

Maintaining and evolving organization control charts has required a significant effort. Without both tangible and intangible results we would not have continued this effort. The organization realizes benefits both before projects begin and while projects are executing.

Root Cause Analysis and Pilot Studies

When process outliers are detected by run rules, the points are quickly investigated. Generally, this is an informal process—a discussion between a project lead and an engineer. If it is a significant, repeated issue, a formal root cause analysis process is performed. This method uses fishbone or Ishikawa diagrams [5], where possible causes for the outliers are listed, followed by discussion and study of likely causes.

In some cases a pilot study or experiment is designed to test whether a change in the process will help prevent or repeat the desired effects in the future. The process is piloted on one or more projects, and data is collected and tested for efficacy. With baseline data, results are empirically compared before and after the process change to determine if it should be institutionalized.

One example of a process change that resulted from such a study was a guideline for review sizes. As review sizes got larger disproportionately fewer defects were identified. The guideline was to keep reviews at or below a certain size, where possible, by not bundling too many modules together. Removing defects as early in the process as possible in a lifecycle has been shown to be more cost-effective [6]. Even without the economic savings, removing as many defects as possible is particularly important for mission-critical software.

Project Estimation

The use of the data collected and analyzed is not restricted to active projects. It also helps estimate future projects. Many companies have a rough idea how long it takes to deliver an average requirement for an average project. Our historical results provide much better estimators for each lifecycle phase and for specific project types. As discussed earlier, organizational control charts provide estimators by lifecycle for avionics versus other types of projects, embedded versus non-embedded, particular customers and project types, and so on.

Since the organizational control charts provide both an average and the variability around the average, project estimates in the form of confidence intervals are produced. So, rather than just estimating a project cost, it is possible to estimate that the project cost has a 85% chance of being more than a low estimate and less than a high estimate. In a competitive marketplace, this can provide a significant edge in winning contracts and with profitability.

These more accurate and precise estimates also provide an advantage in scheduling and resource utilization. Keeping projects reasonably staffed helps avoid engineer burnout, and keeps stakeholders happy by delivering projects on time with high quality.

Another use of the data provided by organizational control charts is in building predictive models. One such model allows more precise predictions of later lifecycle effort, rework hours to fix defects, based on early lifecycle effort, review time and number of defects found. If it is taking longer to review and there are more defects, then the number of hours it will take to rework defects can be estimated. The corollary is also useful in knowing that a project will require less rework time. Having this knowledge helps with resource planning and allows mid-course corrections to help projects stay on schedule.

Conclusion

Overall, SPC has had several positive benefits for the work our team performs. The data provides team leaders the ability to make corrections in real time. This increases the likelihood of achieving the project goals. In other words, utilizing SPC can detect and correct issues before it is too late and encourage repeating desired process improvements.

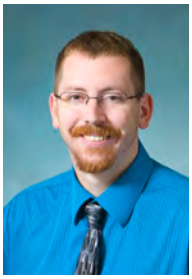
Using control charts also helps identify significant process improvements. Analyzing before and after data helps determine if improvements are effective.

The organizational control charts have allowed us to very precisely understand performance levels. We understand how long tasks take, defect rates, how much rework can be expected, the variability in the complexity of requirements, and more. This is a significant factor for estimating contracts, as well as keeping projects on time, on budget and of high quality.

As has been described above, deploying SPC into an organization is not an easy overnight process. The following actions will help with effective preparation and management of SPC:

- Institutionalize software processes, so results from historical projects can be generalized to future projects.
- Create strong data collection systems that do not burden engineers with record keeping. In our case, these systems actually make an engineer's job easier. We already invested in these systems, so it was not as significant an effort as starting from scratch. Our data collection system was internally produced and has been refined over the last 15 years to satisfy our organizational and customer needs. Companies just getting started in data collection, may be able to find adequate COTS systems that meet their needs.
- Understand the business to determine which key parameters are actually useful to track to help optimize the business and processes.
- Study outliers to discover how and why their processes are producing them. Following this, the organization must try to prevent undesirable events and repeat desirable events by modifying their process.
- Finally, update the organizational control charts as processes and technologies evolve.

ABOUT THE AUTHORS



Craig Hale is a process improvement manager at Esterline Control Systems – AVISTA. He was a lead member of the company's successful CMMI Maturity Level 5 appraisal effort, and he serves as the software engineering process group lead to ensure continuous improvement. Hale holds an Associate degree in computer programming and systems analysis from Southwest Technical College. He is a certified Project Management Professional, and a member of SEI.

Esterline Control Systems – AVISTA
1575 E Business Highway 151
Platteville, WI 53818-3844
Phone: (608) 348-8815
E-mail: craig.hale@esterline.com

Mike Rowe is a professor of software engineering at the University of Wisconsin-Platteville and also works for Esterline Control Systems – AVISTA. He first used statistical process control in the early 1980s to study test yield rates on the HARM missile while with Texas Instruments. Rowe holds two Ph.Ds.—one in physiological psychology, with a minor in statistics, from the University of North Dakota, and the other in computer science from the University of North Texas.

Esterline Control Systems – AVISTA
1575 E Business Highway 151
Platteville, WI 53818-3844
Phone: (608) 348-8815
E-mail: mike.rowe@esterline.com

REFERENCES

1. Chrissis, Mary Beth, et al, CMMI for Development (3rd Edition), Westford, MA.: Pearson Education, Inc., 2011.
2. Western Electric Company, Statistical Quality Control Handbook. (1st Edition), Indianapolis, Indiana: Western Electric Co., 1956.
3. Nelson, Lloyd S., "Technical Aids," Journal of Quality Technology 16, no. 4, (October 1984): pages 238-239.
4. RTCA, Inc., "Software Considerations in Airborne Systems and Equipment Certification – RTCA/DO-178B," Washington, D.C., 1992.
5. Ishikawa, Kaoru (Translator: J. H. Loftus), Introduction to Quality Control, Tokyo: 3A Corporation, 1990.
6. Boehm, Barry W., Software Engineering Economics, Englewood Cliffs, N.J.: Prentice-Hall, 1981.

CALL FOR ARTICLES

If your experience or research has produced information that could be useful to others, CrossTalk can get the word out. We are specifically looking for articles on software-related topics to supplement upcoming theme issues. Below is the submittal schedule for three areas of emphasis we are looking for:

The End of the PC
July/Aug 2012 Issue
Submission Deadline: Feb 10, 2012

Resilient Cyber Ecosystem
Sept/Oct 2012 Issue
Submission Deadline: Apr 10, 2012

Virtualization
Nov/Dec 2012 Issue
Submission Deadline: June 10, 2012

Please follow the Author Guidelines for CrossTalk, available on the Internet at <www.crosstalkonline.org/submission-guidelines>. We accept article submissions on software-related topics at any time, along with Letters to the Editor and BackTalk. To see a list of themes for upcoming issues or to learn more about the types of articles we're looking for visit <www.crosstalkonline.org/theme-calendar>.

High Maturity Pays Off

It is Hard to Believe Unless You Do It

Girish Seshagiri, Advanced Information Services Inc.

Abstract. Is high maturity worth it? Yes, if executive management sponsors the long-term process improvement initiative with constancy of purpose and makes quality the number one goal. We provide a business owner's view of high maturity. We provide hard data on high maturity's impact on customer satisfaction, company profitability, and business strategic decision making as well as intangible results such as self-directed teams, low staff turnover and joy in work. We show that with higher maturity (CMMI® level 5), our processes came under statistical control, and we were able to create a strategic business model with competitive game changers such as firm fixed price contracts and performance guarantees including lifetime warranty on software defects. To the skeptics, we say, "It is hard to believe, unless you do it."

1. Background

Advanced Information Services Inc. (AIS) is in the software application development business. I founded AIS in Peoria, Illinois in 1986 to fill a niche in the market place by creating a workforce with skills in emerging modern programming languages such as C, C++ and Visual Basic and programming paradigms such as object oriented programming. My business objectives were to grow and be profitable while creating much needed high wage, high technology middle class jobs in the U.S. heartland.

I brought to the business a proven track record of successfully managing technical teams in a large corporation. I was a practitioner of the principles of leadership, management and quality articulated by Philip Crosby [1], Edwards Deming [2], and Peter Drucker [3].

2. The Early Years

In the early years of AIS, the company was not profitable because our projects were not predictable. Most of our projects were firm fixed-price contracts. Schedule and budget overruns were normal outcomes. People worked long hours, and heroic efforts were needed to complete projects. Although quality was not measurable, the significant amount of rework necessary at the end of a project was a clear indication of our quality problems and the resulting customer dissatisfaction.

3. The Improvement Initiative

I realized that we had to change the way we managed the software work. What we needed was constancy of purpose with quality as the number one goal. In 1992, I attended a conference on software process improvement based on Watts Humphrey's Managing the Software Process [4]. I returned from the conference convinced that I should sponsor a long-term process improvement initiative to achieve these goals:

- Improve profitability and customer satisfaction by delivering nearly defect free products on predictable cost and schedule.
- Provide a continuing management focus on the progress and visibility of each project from initial commitment to orderly progression through the development lifecycle phases and customer acceptance.
- Continuously improve the software development process through a changed organizational culture biased towards rapid implementation of many small incremental improvements as opposed to a few large changes.

I communicated this vision to everyone in the organization.

4. The Improvement Strategy

We named the initiative Continuous Process Improvement (CPI). We chose the CMM® as the process maturity framework to improve organizational process capability and IEEE standards as the guidelines for software engineering. We were the early adopters of the Personal Software Process (PSP) as the enabling technology to improve individual engineer performance and productivity [5].

We utilized a simple and effective mechanism of Process Improvement Proposal (PIP) to gradually evolve process maturity and ensure company-wide participation in the process improvement journey across maturity levels. We established a Software Engineering Process Group (SEPG), utilizing the skills and experience of many engineers, all on a part-time basis, to evaluate and implement the PIPs. Additionally, we used Watts Humphrey's Managing the Software Process [4] book as a guide and to establish a common vocabulary and communication means.

Our approach was to:

- Conduct self-assessments with a focus on action to achieve measurable results for rework reduction, early defect removal, improved customer satisfaction and predictable outcomes for schedule, cost and quality.
- Maintain organization awareness of improvement efforts through quarterly status reviews.
- Improve continuously and forever.

5. Tracking the Improvement: the AIS Balanced Scorecard

We used the Balanced Scorecard (BSC) method [6] to communicate strategy to the entire organization, link individual accountabilities to strategic objectives, and provide a method to systematically measure progress. The BSC approach identified what we could measure earliest in the software development process which in turn impacts the results later in the process. We then linked the process performance metrics to business objectives for

shareholder, customer, and employee satisfaction. We constructed a cause and effect hypothesis: "When individuals follow the PSP, they will develop work products with targeted percent of defects removed before peer review which will lead to work products with zero post development defects as well as work products with less than or equal to targeted rework effort thereby achieving the strategic internal business process objective—individuals achieve the highest possible quality in their work products. This in turn helps project teams deliver nearly defect free product on time which leads to delighted customers which in turn helps us achieve profitability, revenue growth and joy in work."

We made the SEPG responsible for gathering, analyzing and reporting the BSC measurements in quarterly status review meetings that I chaired.

6. The Results 1992-1999

6.1 The Three Eras

We map the results in three distinct eras corresponding to the changes in the organization's process maturity due to the improvement initiative during 1992- 1999. The first era was the time period before the start of the CPI initiative in 1992 (pre-model era). In the second era, from 1992 to 1995, the focus of the initiative was to stabilize the organization's project planning and tracking processes and implement rigorous requirements engineering and change management processes (CMM-only era). The third era began in 1995 when the initiative focused on improving individual engineer performance and productivity (CMM+PSP era).

6.2 Schedule and Effort Predictability

Figures 1 and 2 graphically depict the impact of the CPI initiative on projects' schedule and effort commitments. Each data point represents a new development phase using the date when the project development phase started, as the horizontal coordinate. The process limits are calculated using a moving range. The process limits in Figures 1 and 2 show the dramatic change in capability to predict schedule and effort from 1988 through 1998.

The data indicate that the average schedule deviation improved from 112% in the pre-model era to 41% in the CMM era to 5% in the CMM+PSP era. While improving the schedule performance met the customer's needs, much of the work was contracted on a fixed-cost bid and, if effort was not predictable, the project phase was not profitable. When AIS began using the PSP, effort became more predictable because of the quality practices the PSP-trained software engineers applied. The data reflect the average effort deviation improved from 87% in the pre-model era to 37% in the CMM era to -4% in the CMM + PSP era.

6.3 Quality and Productivity

PSP-trained engineers working in a mature process improvement have few or no acceptance test and usage defects. The consequent near elimination of rework time and effort flows directly to the company bottom line. A useful metric is System Test duration measured in number of days per thousand lines of code (KLOC). Figure 3 shows the dramatic reduction in system test duration in the CMM+PSP era.

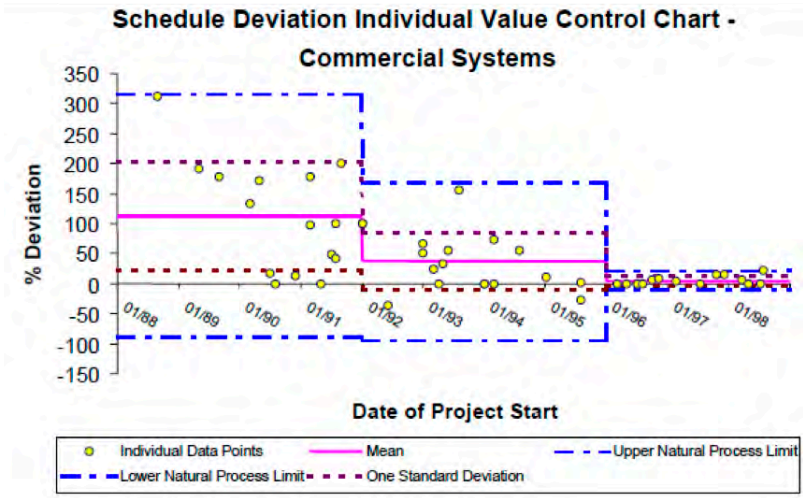


Figure 1: Impact of CPI initiative on projects' schedule deviation

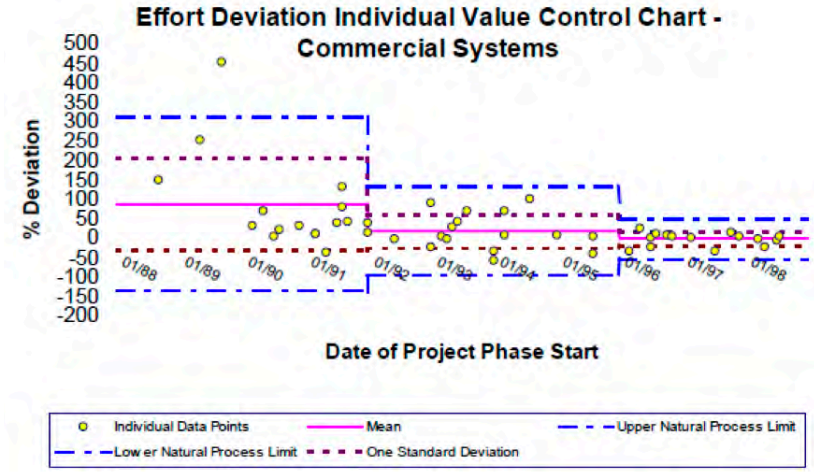


Figure 2: Impact of CPI initiative on projects' effort deviation

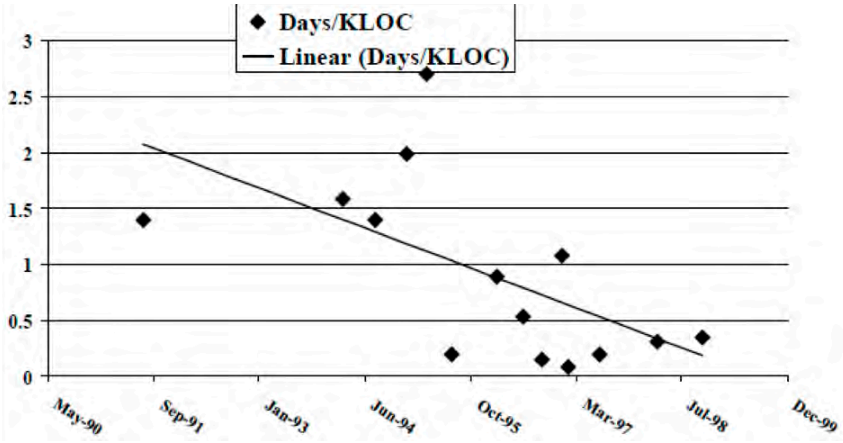


Figure 3: Projects' system test duration measured in number of days per KLOC

It is well known, though seldom practiced, that it is cheaper to remove defects earlier in the lifecycle phases such as requirements, design, and coding than later phases such as integration, system, and acceptance test. A useful metric is the percent of defects removed prior to test. AIS teams of PSP-trained engineers consistently deliver very high quality product into test. Figure 4 shows that in the CMM+PSP era, AIS projects through the consistent use of mature processes such as team inspections and personal reviews removed more than 75% of defects prior to test.

6.4 Project and Company Profitability

In the era before 1992, AIS was profitable in only one out of the five years. Individual project profitability and overall AIS revenue and profits improved significantly as the CPI initiative progressed. In the CMM only era during the years between 1992 and 1995, profit as a percentage of revenue averaged 5.7%. In the CMM+PSP era, profit as a percentage of revenue averaged 9.9% primarily due to reduction in test and rework. These gains occurred at the same time that the AIS organization was experiencing growth from 21 people in 1990 to over 140 people by the end of 1998.

6.5 Peer Recognition

In 1999, the software professional community recognized the AIS engineers and managers with the IEEE Computer Society Software Process Achievement Award, which is similar to the Malcolm Baldrige National Quality award [7]. In addition, the business community recognized AIS with a Blue Chip Enterprise Initiative award given by the U.S. Chamber of Commerce and Mass Mutual. The awards recognized the impact of the CPI initiative in significantly improving AIS financial performance.

7. Holding the Gains

The initiative's results convinced the entire organization that quality had the biggest impact on schedule, cost and profitability and quality work was more predictable. At the same time, we recognized that in order to hold the gains, we needed to focus on training, teamwork, and making disciplined commitments while meeting customers' increasingly demanding time to market needs.

Our actions included:

1. Implemented a Software Engineering Certificate program and required that all AIS engineers and managers must complete the following courses:
   a. Requirements engineering and management
   b. Software Inspections
   c. PSP for Engineers
   d. Managing the Software Process
   e. Managing Technical People
2. Piloted the Team Software Process (TSP) during 2000 -2001 [8].

8. Making Disciplined Commitments

After the successful pilot, AIS adopted the TSP. Software project teams utilize the multi-day team launch mechanism of the TSP to ensure that teams tailor the AIS CMMI Level 5 process and create detailed granular project plan containing hundreds of individual tasks. AIS policy requires that all team members participate in the team launch. Team members with the help of a TSP coach, estimate effort and schedule based on personal historical data. The teams successfully negotiate an aggressive and realistic schedule with the stakeholders.

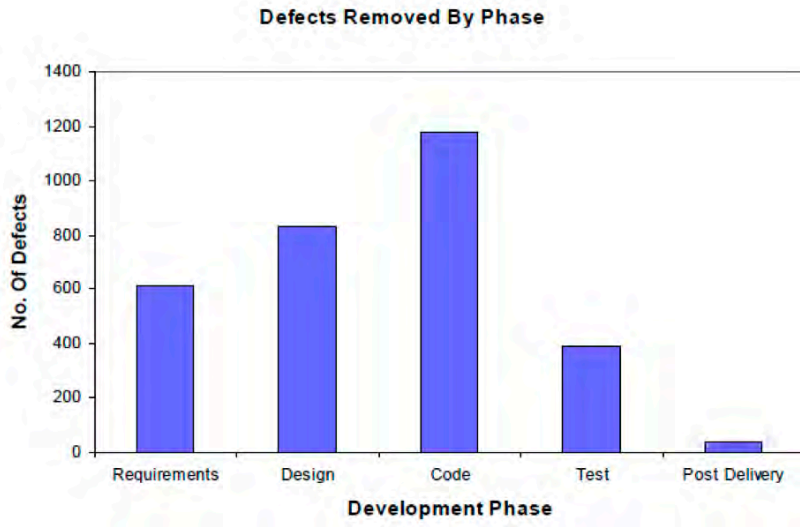


Figure 4: Defects removed by lifecycle phase reported by projects 1995 - 1998

9. The Results 2002 - Present

9.1 CMM/CMMI Assessments

Table 1 shows the history of AIS's CMM/CMMI assessments and the progression to CMMI Maturity Level 5. AIS is only one of 18 organizations and the only small business in the U.S. assessed at CMMI Maturity Level 5 [9].

Date	Levels Assessed	Levels Satisfied	Assessor
Nov 2002	SW-CMM Levels 2 to 4	3	External
Nov 2004	SW-CMM Levels 2 to 4	4	Internal
Dec 2005	SW-CMM Levels 2 to 5	5	Internal
Dec 2007	CMMI Maturity Levels 2 to 5	5	External
Dec 2010	CMMI Maturity Levels 2 to 5	5	External

Table 1: History of AIS CMM/CMMI Assessments

9.2 Understanding What We Do and How We Do It

The CMMI Level 5 process has given us a greater understanding of how the organization does the software application development work, the capability of the organization's defined processes and sub-processes, the ability to analyze the common and special causes of variation and institutionalize defect prevention activities. Such an understanding has helped the organization make business decisions such as firm fixed price contracts, offer performance guarantees, and provide greater value to customers.

9.3 Changed Organizational Culture

AIS engineers submitted more than 1,400 individual PIPs and evolved the AIS process from SW-CMM Level 1 to Level 5, and then to CMMI Maturity Level 5. The AIS staff is to be commended for broad based involvement in continuous improvement of the AIS defined process and achieving measurable results for rework reduction, early defect removal, improved customer satisfaction and predictable outcomes for schedule, cost and quality. One of the benefits of a high maturity workforce is what we call Level 5 behavior that ensures that the organization will improve continually and forever because of its people.

9.4 Self-managed Teams

The TSP practices assisted by a coach enabled AIS project teams to jell at project initiation and to manage themselves throughout the project duration. The external CMMI appraiser identified the following organizational strengths in the appraisals conducted in 2007 and 2010:

- TSP coaches provide continuous mentoring for project team members.
- Process focus at all levels of the organization.
- Open communication.
- Self-managed team structure and roles.
- Individuals with:
  - Strong quality focus
  - Commitment to customer and organization
  - Sense of ownership
- Opportunity for involvement with multiple groups within the organization.
- Empowered to make decisions that affect the organization.

9.5 Processes Under Statistical Control

Project teams utilize statistical process control charts to analyze variation in results for schedule, effort, inspections, and defects. The teams supported by the SEPG use the charts to analyze the common and special causes of variation and verify the impact of changes to the process.

The following charts are from the SEPG presentation in a quarterly status review meeting in December 2010.

9.6 Predicting Quality

TSP teams make detailed quality plans during the TSP team launch. The quality plan is based on historical data on defect injection rates, and removal yields by lifecycle phase. Figure 9 shows the planned vs. actual defects removed by lifecycle phase. This data is valuable in helping teams determine whether or not they are meeting the quality plan. Teams use the data to report weekly the estimated number of defects in systems and user acceptance test and take corrective action, if needed, to meet project's schedule and effort commitments.

One of the more significant payoffs from high maturity is the ability to predict if a component is likely to have defects in downstream integration, system, and user acceptance testing. It is extremely useful to know which components are likely to be error-prone so that we can be pro-active and take corrective action.

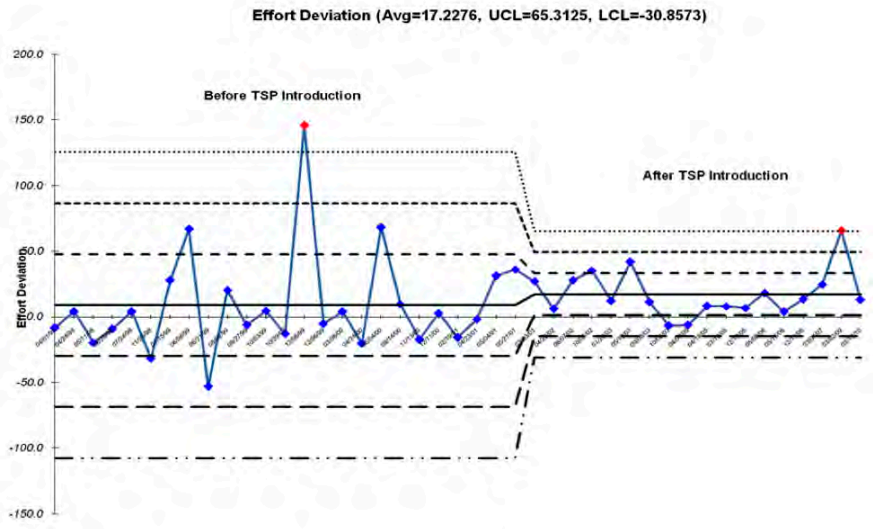


Figure 5: Impact of TSP on projects' effort deviation

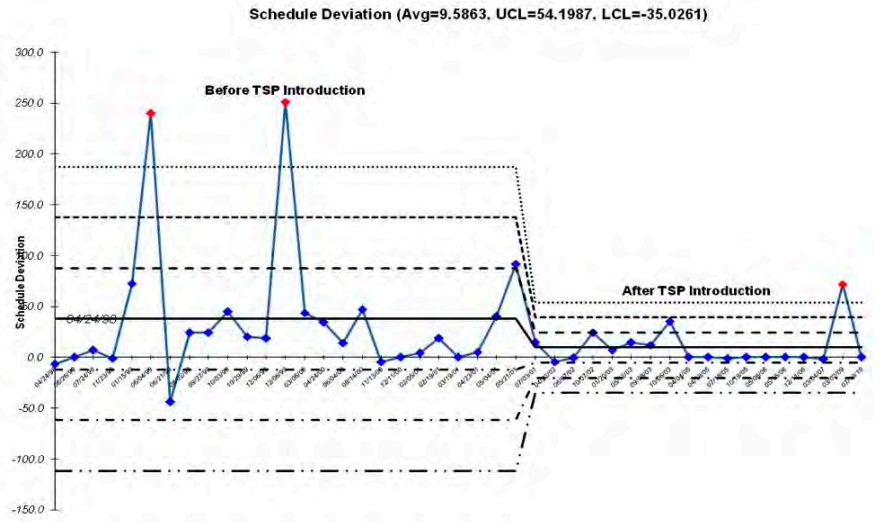


Figure 6: Impact of TSP on projects' schedule deviation

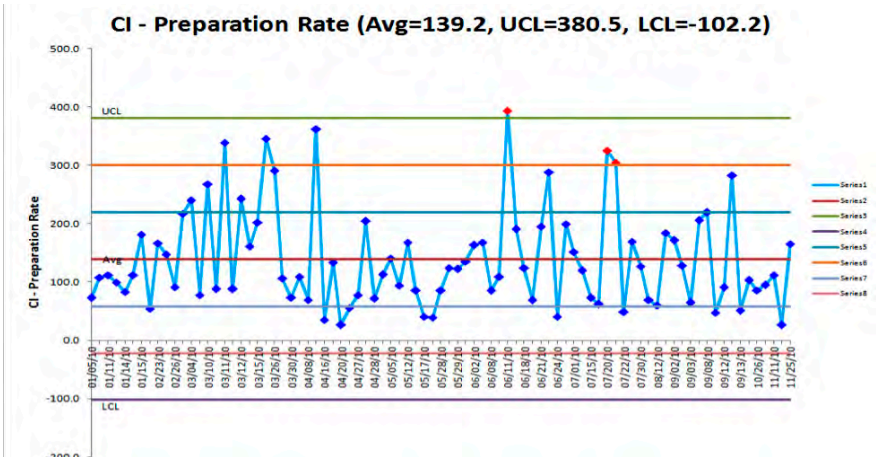
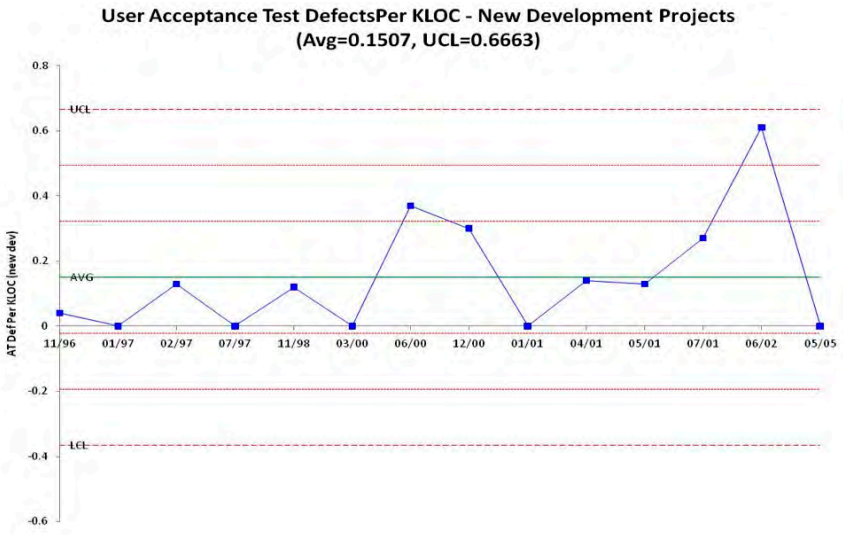


Figure 7: Ensuring effectiveness of peer review (inspection) process



Figures 8: Defect density of new development projects

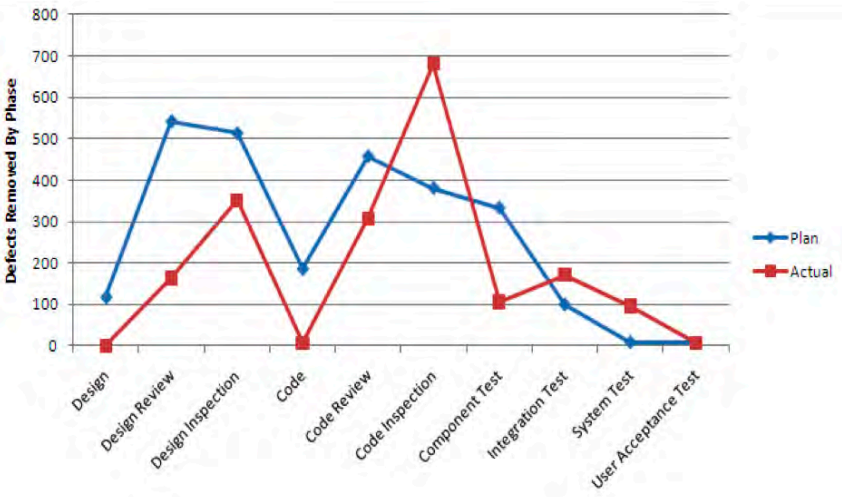


Figure 9: Planned vs. actual defects by lifecycle phase reported by a project in 2009

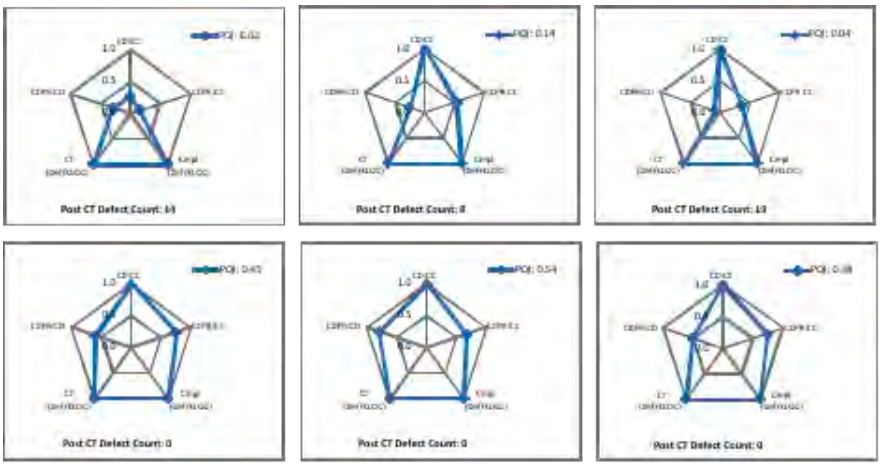


Figure 10: Component quality profile and PQI

The TSP uses a Process Quality Index (PQI) metric to predict whether components that have been unit tested will have downstream defects in integration, system and customer acceptance testing. The PQI is calculated by multiplying together five components of the quality of a component to give a value between 0.0 and 1.0. Experience to date shows that, with PQI values above about 0.4, components typically have no defects found after development. By plotting five quality measures on radar charts (as shown in Figure 10), a component's potential quality problems can be identified in time to take corrective action.

Figure 10 shows a real life example from an AIS TSP team project. The components that had PQI less than 0.4 had integration and system test defects while the components that had PQI greater than 0.4 had zero defects in integration and system defects.

9.7 Scaling Up to Larger Teams, Ensuring Zero Vulnerability and Low Staff Turnover

With processes and sub-processes under statistical control, AIS is able to field larger team sizes and maintain schedule, cost, and quality within known AIS process capability. An AIS team of 17, recently delivered more than 500,000 lines of VB.Net and SQL code on time to a federal agency on a firm fixed price contract. To ensure compliance with the Federal Information Security Management Act, we conducted two independent tests to detect vulnerability in the code. Both tests reported zero vulnerability. Such unprecedented quality performance is due to high maturity practices and the constancy of purpose of the AIS staff with quality as the number one goal. There has been no staff turnover on this project since 2008. For about half the team members, this was their first job straight out of college. We are confident we can scale up our high maturity practices up to 1 million lines of code developmental effort with a team size greater than 25 and maintain AIS historical averages for schedule, cost and quality performance.

9.8 Voice of the Customer

When AIS teams complete a project phase, the AIS SEPG conducts project phase review with customer input on whether the AIS team met or exceeded customer expectations and needs for quality, value, and timeliness. Table 1 shows a summary of the SEPG reports in quarterly status review meetings since 2001 indicating that AIS teams consistently meet or exceeded customer needs and expectations.

Percent customer responses indicating	Quality	Value	Timeliness
AIS team met or exceeded needs and expectations	90.7	95.6	90.4
AIS team needs to improve	9.3	4.4	9.6

Table 2: Customers' phase review responses

9.9 Performance Metrics That Matter

AIS averages for performance metrics that matter are superior to industry averages (Table 2). Customer benefits include significantly less time in acceptance test (agility), and lifetime warranty on defects found in production use (quality). In AIS projects, cost of finding and fixing bugs is no longer the number one cost driver in software development. Customer and AIS staff have more time for new features, enhancements, and technology solutions (innovation). Reduced rework and predictable development schedules lead to work/life balance. High performance jelled team environment leads to zero to low staff turnover (joy in work).

Performance Metrics That Matter	Industry Average	AIS Average
Schedule deviation	>50%	<11%
No. of defects in delivered product 100,000 LOC	>100	<15
% of design and code inspected	<100	100
Time to accept 100,000 LOC product	4 months	5 weeks
% of defects removed prior to system test	<60%	>85%
% of development time fixing system defects	>33%	<10%
Cost of quality	>50%	<35%
Warranty on products	?	Lifetime

Table 3: Benchmarking performance metrics that matter

10. The Future

10.1 Business Strategy Based on Excellence

AIS is now able to align its business strategy with the federal government's move to a larger percent of firm fixed price performance based contracting. Based on the performance of AIS teams during the past 10 years, we now offer guarantees for cost, schedule, agility and quality (Table 4).

Cost
Firm fixed price upon acceptance of requirements specifications
Schedule
Not to exceed 10% of committed schedule
Weekly status reporting with ability to detect as little as one-day schedule slip
Agility
Time in test significantly less than customer's historical average
Rework time significantly less than customer's historical average
Quality
Acceptance test defects significantly lower than customer's historical average
AIS *will fix defect found in production use free for the life of the product!!!*

Table 4: AIS performance guarantees

10.2 Joy In Work: the Ultimate Pay-off

High maturity by itself may not provide sustaining competitive advantage in the knowledge economy. Jelled teams that are self-directed and find joy in work will continue to out perform industry averages for knowledge work.

10.3 High Maturity: Not the End, but the Beginning

For us, high maturity is not an end, it is just the beginning in helping us understand the leadership challenge in the knowledge economy, make the connection between agility, quality, innovation, joy in work, profits and human values, and begin transformation to a life of greatness.

It is Hard to Believe, Unless You Do It

We conclude with our belief that constancy of purpose and making quality the number one goal are more important than the specific methods or models used. We realize that for many of the skeptics, no amount of data will convince them that high maturity is worth it. To them, we say, "It is hard to believe, unless you do it."

Disclaimer:

CMMI® and CMM® are registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

ABOUT THE AUTHOR

Girish Seshagiri is CEO of AIS, a winner of the IEEE Computer Society Software Process Achievement Award. AIS is one of the very few organizations in the U.S. whose software process capability is assessed at SEI CMMI Level 5. Seshagiri has more than 30 years experience managing technical teams. He has an MBA degree from Michigan State University.

Girish Seshagiri
Phone: 703 426-2790
E-mail: girish.seshagiri@advinfo.net

REFERENCES

1. Crosby, Philip B., "Quality is Free", McGraw-Hill, 1980.
2. Deming, W. Edwards, "Out of the Crisis", MIT Press, 1982.
3. Drucker, Peter F., "The Effective Executive", Harper Colophon, 1985
4. Humphrey, Watts S., "Managing the Software Process", Addison-Wesley, 1989.
5. Humphrey, Watts S., "A Discipline for Software Engineering" Addison-Wesley, 1995.
6. Kaplan, R., Norton, D., "The Balanced Scorecard", Harvard Business School Press, 1996.
7. IEEE Computer Society, " IEEE Computer Society Awards ", <http://www.computer.org/portal/web/awards/processachievement>
8. Humphrey, Watts S. "The Team Software Process", Technical Report, CMU/SEI- 2000-TR-023, ESC-TR-2000-023, Nov, 2000.
9. Wilson, Hal., "Publicly reported data begging for analysis", CMMI Technology Conference, November 1999.

Why CMMI Maturity Level 5?

Michael Campo, Raytheon Company

Abstract. Most organizations that use the CMMI® stop their process improvement journey at Maturity Level 3 or less. Yet, the CMMI high maturity processes offer the greatest potential for ROI. This article outlines why the high maturity process areas have the highest ROI potential, and presents data from Raytheon Integrated Defense Systems (IDS), a CMMI Maturity Level 5 organization, as an example.

Introduction

CMMI high maturity levels have always been controversial, highlighted by cost vs. benefit debates, supplier selection, and model interpretation issues. "It costs too much!", "Where are the benefits?", "Our customer only wants us to be Level 3!" are typical comments heard. Many organizations decide that Maturity Level 3 is adequate, and choose not to pursue high maturity. Over 90% of organizations submitting appraisal results to SEI are Maturity Level 3 or lower [1]. Lost amidst the rhetoric is why an organization would want to be Maturity Level 5 in the first place. This article demonstrates that Level 5 can be the most cost effective of all the maturity levels. IDS will be used as an example to depict the value, benefits, and impact of implementing CMMI Maturity Level 5 processes.

History and Controversy

High maturity controversy dates back to the CMM® years. When the CMM was released, few prototypes of high maturity organizations existed. The most oft-cited example at the time was the Space Shuttle Onboard Software project at IBM-Houston [2].

The number of CMM high maturity projects increased after the 1999 publication of a memo from the Office of the Undersecretary of Defense (often referred to as "The Gansler Memo") set an expectation of CMM Maturity Level 3 for software development contractors of ACAT I programs. If CMM Maturity Level 3 was an expectation, then CMM Maturity Level 5 was seen as a key discriminator for winning contracts. This pattern continued with the release of CMMI. If CMM Maturity Level 5 was a discriminator, then CMMI Maturity Level 5 was seen as an even greater discriminator.

Unfortunately, customer satisfaction did not appear to rise in parallel with CMMI high maturity level ratings. Many customers lamented that they were not seeing what they expected as high maturity results on programs from high maturity organizations.

This led to a series of high maturity level-setting clarifications and standards from the SEI. New courseware was developed called "Understanding CMMI High Maturity Concepts." A Standard CMMI Appraisal Method for Process Improvement (SCAMPI)℠ High Maturity lead appraiser certification program was established. Audit criteria for high maturity appraisals were released. Numerous SEI presentations delivered at conferences addressed perceived high maturity misinterpretations, and stressed the importance of the CMMI informative material in understanding and implementing high maturity practices. Fixing high maturity became the impetus behind the November 2010 release of CMMI V1.3.

The Big Picture

Looking at the big picture, it becomes obvious that the high maturity controversy is really a high maturity appraisal controversy. Let us remember that the CMMI was created to support business improvement. As a model containing best practices, the CMMI is a strategic tool used to help achieve business objectives. Those business objectives are expected to be achieved from improved performance, not through the marketing of maturity levels. The CMMI model and its maturity levels are a means to an end, not an end unto themselves.

Any organization using the CMMI to improve business performance can open the model and select related good ideas contained therein regardless of whether those ideas are described in bold font (required and expected material) or normal font (informative material). The "required, expected, or informative" designation of CMMI material only becomes relevant in appraisals. Although CMMI V1.3 restructured some of the material contained in the high maturity process areas, and clarified high maturity related glossary definitions, the essential high maturity concepts remained unchanged from CMMI V1.2. Organizations that took a holistic approach to implementing CMMI V1.2 high maturity will see little change in CMMI V1.3. Organizations that took an appraisal-centric approach to CMMI V1.2 high maturity, focusing solely on required and expected material, are likely to see significant change in CMMI V1.3 high maturity.

The practices described in CMMI Maturity Level 2-5 process areas all offer benefits.

CMMI Maturity Levels 2 and 3 focus on disaster prevention and gaining control of the way work is performed in an organization:

- Maturity Level 2 processes focus on disaster prevention due to unrealistic plans, lack of requirements management, poor configuration management and quality, management without measures, and ineffective supplier management.
- Maturity Level 3 processes focus on increased consistency of performance using common organizational processes tailored by individual programs, and increasingly proactive management techniques.

CMMI Maturity Levels 4 and 5 offer a much more strategic focus. This focus is built around establishing and managing against quality and process performance objectives that align with business objectives:

- Maturity Level 4 processes establish quality and process performance objectives that trace directly to business objectives.

The organization develops a statistical understanding of its ability to perform against the quality and process performance objectives by using process performance baselines and models. The quality and process performance objectives are flowed down to individual programs that manage against those quantitative targets. In this manner, individual programs in the organization recognize their role in business success, and take action accordingly if the objectives are not being met.

Maturity Levels 2 and 3	Maturity Levels 4 and 5
PMC SG1: Actual project progress and performance are monitored against the project plan.	OPP SP 1.1: Establish and maintain the organization's quantitative objectives for quality and process performance, which are traceable to business objectives.
IPM SP 1.5: Manage the project using the project plan, other plans that affect the project, and the project's defined process.	QPM SP 2.2: Manage the project using statistical and other quantitative techniques to determine whether or not the project's objectives for quality and process performance will be satisfied.
OPF SP 1.3: Identify improvements to the organization's processes and process assets.	OPM SG2: Improvements are proactively identified, evaluated using statistical and other quantitative techniques, and selected for deployment based on their contribution to meeting quality and process performance objectives.

Table 1

Raytheon Goals

- Be regarded as a customer focused company.
- Grow revenue faster than market. Build on good performance in improving cash flow. Execute well and with predictability.
- Retain and attract world-class talent while providing superior opportunities for employee development. Treat all employees with respect. Leverage our diversity efforts as a competitive advantage, continuing Raytheon's leadership in diversity.
- Improve ROIC for Raytheon Company. Take R6σ™ to the next level, further engaging customers and partners. Deliver greater value and predictability through IPDS, EVMS, and CMMI®.

Raytheon IDS Engineering Quality and Process Performance Objectives

Cost	Schedule	Quality	People
Cost Performance Index ≥ x	Schedule Performance Index ≥ x	Defect Containment ≥ x%	Average x Hours Training per Employee
Productivity x% > Bid	Productivity x% > Bid	Defect Density < x	
Defect Containment ≥ x%	On time Deliverables Average > x%	Requirements Volatility ≤ x%	
Requirements Volatility ≤ x%			

Figure 1

• Maturity Level 5 processes establish a system of continuous evaluation and maintenance of business objectives, and the associated quality and process performance objectives. Progress against those objectives is analyzed, and process improvements are identified based on their contribution towards achieving the objectives. Causal analysis and resolution techniques are used in support of these activities.

Let us compare some example Maturity Level 2 and 3 processes against Maturity Level 4 and 5 processes. See Table 1 [3].

The Maturity Level 2 and 3 processes are all good things to do: having project plans, managing against project plans, and identifying process improvements. Note, however, the Maturity Level 4 and 5 focus on quality and process performance objectives derived from business objectives. Flowing the quality and project performance objectives down to programs, and using quality and process performance objectives as the basis for process improvement activity, is what sets the stage for the greater return on investment than may be realized from Maturity Levels 2 and 3. A business can only be successful if its programs are successful. At Maturity Levels 4 and 5, the entire organization becomes enlisted in helping the business achieve its objectives. Programs have to manage against those objectives, report status to higher-level management regularly, and take actions when the objectives are not being achieved. Programs in turn may establish their own quality and process performance objectives, based on achievement of award fees or other significant results.

Raytheon IDS Engineering Quality and Process Performance Objectives

Raytheon, like many large organizations, annually establishes high-level goals for the company. See Figure 1. The Raytheon goals are business objectives designed to help the company be successful. Within Raytheon IDS, engineering evaluates the Raytheon goals and establishes quality and process performance objectives based on what engineering must accomplish to help the company achieve its goals. As in many organizations, the quality and process performance objectives relate to cost, schedule, quality, and customer satisfaction. See Figure 1. Those objectives are flowed down to programs as performance management targets, and to the organization as process improvement prioritization criteria. In this manner, the entire organization becomes enlisted in a "grass roots" effort to help Raytheon achieve its goals.

Raytheon IDS ROI from CMMI Maturity Level 5

In November 2008, Raytheon IDS was appraised to be CMMI Maturity Level 5 for Systems, Software, and Hardware Engineering. Previously, portions of Engineering had achieved Maturity Level 3 in 2003 and Maturity Level 4 in 2005. In 2009, a CMMI Maturity Level 5 return on investment analysis was performed. The data used in this study compared Raytheon IDS Systems, Software, and Hardware Engineering performance in 2005 versus 2008. Data on cost, schedule, quality, and customer satisfaction were analyzed, and overall ROI determined based on investment and savings.

• Investment was defined as the cost of all activities to incorporate Maturity Level 4 and 5 practices into our processes

and be appraised. This includes development and deployment of updated processes and enablers (e.g., process performance models), training, and all appraisal costs.

• Savings was calculated by applying 2005 baseline rates (e.g., 2005 productivity) against 2008 size (e.g., lines of code) to determine a "projected cost at baseline rates," and comparing the "projected cost at baseline rates" to 2008 actual costs.

Individual Causal Analysis and Resolution (CAR) and Organizational Innovation and Deployment (OID) improvement activities were examined. CAR and OID activities are process improvements made specifically to help achieve quality and process performance objectives. Examples of these improvements included increasing peer review effectiveness, steps taken to reduce requirements volatility, and deployment of improved software static analysis tools. As noted in Table 2, the overall CAR and OID ROI was 38.4:1. The large ROI is not surprising, given that these activities are focused on achieving quality and process performance objectives related to improving productivity, defect containment, and similar high-yield initiatives.

	Total ROI	Highest ROI	Lowest ROI	Median ROI	Number of data points
Total CAR/OID ROI	38.4:1	183.3:1	1.9:1	14.3:1	19
ROI on OID Projects	57.1:1	183.3:1	10.7:1	50.8:1	5
ROI on CAR Projects	25.8:1	85.5:1	1.9:1	9.6:1	14

Table 2

System, Software, and Hardware Engineering performance improvements on targeted tasks are listed in Table 3.

Engineering Discipline	Performance Improvement
Systems Engineering	56% Requirements Volatility improvement 14.3% Requirements Development Productivity improvement 4% Cost Performance improvement 63% variance reduction in Cost Performance Index
Software Engineering	65% Design-Code-Test-Integration Productivity improvement 11.6% Defect Containment improvement
Hardware Engineering	25% Mechanical Engineering Productivity improvement 33% Analog Electrical Design Productivity improvement 56% Digital Electrical Design Productivity improvement 65% Drawing Checking Defect Density improvement
All	On time Deliverables > 99% since 2006

Table 3

Raytheon IDS Engineering realized an overall ROI of 24:1 from CMMI Maturity Level 5 activity. The 2006 SEI study "Performance Results of CMMI-Based Process Improvement" [4] showed CMMI ROI ranging from 1.7:1 to 27.7:1, with a median of 4:1. The Raytheon IDS Engineering ROI would place near the top of that scale.

The Raytheon IDS Engineering ROI and performance results are a direct consequence of meaningful process improvement aligned with the business objectives and associated quality and process performance objectives. The high maturity focus on improving processes that have the most impact on achieving those objectives (e.g., productivity, defect containment, requirements volatility), produced results that added value to the business. This is the essence of high maturity.

The high maturity benefits described in this article are based on a Raytheon IDS study performed in 2009. A 2010 follow-on study showed that benefits from CMMI Maturity Level 5 continue to accrue. CMMI Maturity Level 5 remains a cornerstone of the Raytheon IDS business strategy today.

Summary

What an organization gets out of CMMI-based process deployment and appraisals is a function of what the organization puts into it. Organizations that focus on maturity level ratings and CMMI minimal compliance are unlikely to derive benefits from their investment. Organizations that use the high maturity principles to deploy meaningful process improvement aligned with business objectives are organizations that are much more likely to reap greater return from their investment.

Maturity Levels 2 through 5 all offer benefits. Maturity Levels 2 and 3 help prevent disasters and gain control in the way work is performed in an organization. There is no denigrating the improvements an organization can realize from implementing Maturity Level 2 and 3 processes.

However, Maturity Levels 2 and 3 are not focused on quality and process performance objectives as the driver of process improvement activity, and therefore set a lower ceiling on the benefits of CMMI-based process improvement. Using Maturity Level 4 and 5 processes to manage against quality and process performance objectives creates a grass roots movement within an organization to meet business objectives. An organization where all individuals recognize their role and responsibility for business success is an organization that is more likely to achieve success.

Disclaimer:

ABOUT THE AUTHOR

Michael Campo is a Principal Engineering Fellow at Raytheon Company. As process group leader for Raytheon Integrated Defense Systems, Mike developed and deployed processes that led to achievement of CMMI Maturity Level 3 in 2003, Maturity Level 4 in 2005, and Maturity Level 5 in 2008. Mike is a member of the CMMI V1.3 Core Model Team, the CMMI V1.3 Training Team, the CMMI Configuration Control Board, and the NDIA CMMI Working Group.

Michael Campo
Raytheon Company
50 Apple Hill Dr
Tewksbury, MA 01876
E-mail: Michael_J_Campo@raytheon.com

REFERENCES

1. SEI. "CMMI for Development SCAMPI Class A Appraisal Results 2010 End-Year Update", SEI, Carnegie Mellon University, March 2011. <http://www.sei.cmu.edu/cmmi/casestudies/profiles/cmmi.cfm>
2. SEI. The Capability Maturity Model, Guidelines for Improving the Software Process v1.1. Addison Wesley, 1994.
3. Chrissis, Konrad, Shrum. CMMI for Development V1.3, Guidelines for Process Integration and Product Improvement. Addison Wesley, 2011.
4. Gibson, Goldenson, Kost. "Performance Results of CMMI-Based Process Improvement." CMU/SEI-2006-TR-004, 2006.

Taking an Agile Organization to Higher CMMI Maturity

Paul E. McMahon, PEM Systems

Abstract. Many believe the CMMI® [1] and Agile methods [2] are at odds. This article will provide practical techniques to take an Agile organization to CMMI Level 3 without jeopardizing its Agile approach. The phrase "Agile organization" as used in this paper refers to any organization that uses an Agile approach on the majority of its projects. By "Agile approach" we mean the extension of Agile software concepts such as iterative development, daily standup meetings, frequent delivery, customer collaboration, and continual refinement of plan, to include systems engineering and project management.

Actual case study data is shared where the techniques have proven successful. The material is based on a case study described in greater detail in the book, "Integrating CMMI and Agile Development: Case Studies and Proven Techniques for Faster Performance Improvement" [3]. The techniques discussed can actually help any organization implement an effective and efficient CMMI effort whether or not the starting point is Agile. The reason an "Agile organization" is emphasized in the paper is because the techniques described are particularly important to these types of organizations in order to allow them to continue to employ their Agile approach and achieve higher CMMI maturity.

Background

BOND is an organization that was started by two retired military men. In 2000 I was asked to conduct a gap analysis using the CMM® model to help BOND initiate a process improvement effort. At that time the company had only 25 people and no documented processes. For the next few years the organization attempted unsuccessfully to move its process improvement effort forward. In 2003 they asked me to conduct another gap analysis, this time using the CMMI model. After this gap analysis they asked me to become more involved helping them move their process improvement program forward. In 2005 BOND achieved a formal CMMI Level 3 in eight of the 18 Process Areas required for a full CMMI Level 3. In 2007 they achieved a formal full CMMI Level 3 in all 18 Process Areas. By the time of the formal appraisal in 2007 the organization had grown to 150 people.

Challenges Faced

In 2003 when we began the process improvement effort at BOND I was given two key challenges by the organizational leaders. First, add the process discipline required to help them manage their continued planned growth. Second, maintain the successful Agile culture which the leaders strongly believed was key to their success. That "Agile culture" included an emphasis on a close collaborative relationship with their customers, early customer demonstrations, daily standup meetings, and frequent product deliveries. The majority of the projects at BOND utilized a Scrum approach tailored to specific project constraints.

In this article I share key practical techniques used at BOND to help them achieve their goal. Specifically I share three keys to conducting a gap analysis against the CMMI model for Agile organizations, and three key tailoring areas found effective in Agile organizations for running Technical Working Groups (TWGs) to develop CMMI Level 3 compliant processes. The added value the CMMI can bring to a previously successful Agile organization is also shared along with an example of a key technique employed to gain the buy-in for needed changes.

Fundamental Guidance Employed at BOND

Contrary to popular belief the CMMI is not a set of dictated practices. It is a process improvement reference model intended to help you ask the right questions leading to the best decisions for your organization. For example, the CMMI leads you to ask, "Do you have sufficient resources on your project?" This question is methodology agnostic, and can help any organization including Agile organizations. But using an Agile approach alone will not lead you to ask this question.

This is an example of how the CMMI can help an Agile organization. On the other hand, Agile concepts provide a wealth of potential "how to" approaches that can achieve the intent of CMMI practices. An example is daily standup meetings. But daily standup meetings may not work in all situations, such as when your team is distributed in different time zones. The CMMI is about "what" you must do. Agile techniques provide "potential" how-to options. This is the fundamental guidance we employed in making key decisions concerning process improvement at BOND.

This article will demonstrate how applying this guidance allowed BOND to maintain the Agile values they were experiencing in 2003 when they achieved their CMMI Level 3 in 2007. It will also provide guidance on what you can do in your organization to effectively integrate the CMMI and Agile development given your specific situation.

Gap Analysis in Agile Organization

A gap analysis is an assessment of an organization based on the CMMI model. The result is a strengths, weaknesses and recommendations report that can be used to plan the road forward toward higher CMMI maturity. When the report is presented to an organization's leaders one should stress that weaknesses identified against the CMMI model are "potential" weaknesses to the organization that we may or may not have to take action to address. Recommendations related to the conduct of a gap analysis in an Agile organization involve three key areas; Gathering accurate data, Reporting results, and Handling "potential" weaknesses. Each is discussed below.

Gathering Accurate Data

There are multiple possible approaches to conducting a gap analysis against the CMMI model. A common approach is to focus on the documentation (existing processes and products produced), and supporting this review with a few interviews. For Agile organizations it is recommended to switch the primary focus to the interviews, and conduct the interviews one-on-one, as opposed to group interviews, which is often done with formal CMMI appraisals. It is also recommended to use no CMMI terms. Ask simple questions and encourage the person being interviewed to just talk about how they do their job. Then listen, and take plenty of notes. The rationale for this approach is based on the fact that our goal is to gain the most accurate picture of how the people operate in the organization today providing a starting point for the process improvement effort.

Reporting Results

It is recommended for gap analysis reports to go much deeper than traditional gap analysis reports and they should be based on specific objective data heard in the interviews, or seen in documentation reviews. The rationale for this is based on the fact that specifics are usually needed for management to buy-in to the changes that the organization requires for higher CMMI maturity. Too often results are raised up to an abstract level due to fear of "attribution." This is understandable, as we do not want findings to be attributed to individuals. However, we have found a more effective way to handle this by only reporting "potential" critical specific patterns that have been uncovered by hearing the information in two or more interviews. This addresses the individual attribution concern, and also helps us achieve buy-in for the value-added changes the organization needs.

Handling "Potential" Weaknesses

Handling "potential" weaknesses can best be described through an example. When you are conducting a gap analysis interview eventually you will get around to the products produced by the worker. Then ask, "Does anyone else look at these products you produce?" Often, in Agile organizations, the answer is, "We do not do formal peer reviews."

When you hear this just note it as a "potential" weakness recognizing we will need to come back later and dig deeper to find out what is really going on. You could alternatively just report it to management as a weakness that needs to be fixed. You could say, I heard you do not do peer reviews, and you need to do peer reviews because they are an expected practice within the CMMI model. While this would be the easiest thing to do, it would also add the greatest risk to the goal of maintaining the successful Agile culture. Later in this paper we will explain how to handle these "potential" weaknesses. Next we discuss the recommendations to move forward after a gap analysis through TWGs.

TWGs

The TWG is the next step after the gap analysis. The traditional responsibilities of a TWG are to develop and document processes, and address weaknesses identified in the previous gap analysis. The members of the TWG should be the subject matter experts that use the processes being developed. There are three areas recommended to tailor the traditional responsibilities of a TWG for Agile organizations; Training processes, rationale for "stretches", and approach to "potential" weaknesses. Each is discussed below.

Training Processes

It is recommended to hold TWG members responsible for training—at least the first round of roll-out training to the organization. The reason for this is because there is no one better able to explain why decisions were made than those who developed the processes during the TWG effort. Too often TWGs are disbanded after they develop the processes, and as a result this all-important training aspect does not get the attention it deserves.

Rationale for "Stretches"

Stretch areas are areas where we are asking personnel in the organization to change their behavior. It is recommended to require TWGs in all process roll-out training to focus on "stretch" areas and always provide the rationale for each stretch. This is done for two reasons. First, it helps to ensure we are mitigating the risk of hindering the existing successful Agile culture. By requiring the TWG to do the training and to provide the rationale for anything in the new processes that is a "stretch", we ensure the TWG members thoroughly think-through what they are requiring in the new processes.

Too often when TWG members think their job ends once they have developed the process documentation they do not take the impact of their decisions on the organization serious enough. It is also recommended to let TWG members know, "because the CMMI says so" is never a valid reason by itself to include a new "stretch." The CMMI is a reference model that helps us ask the right questions leading to the right practices for our organization. It is not a set of dictated practices, which is too often the way the model has been applied in the past.

Approach to "Potential" Weaknesses

The first rule recommended to give TWG members is, "Always start with the intent question." This is a rule I learned from a CMMI lead appraiser I worked with many years ago. What she meant was, whenever you are dealing with a potential weakness, ask yourself, "What is the intent of this practice in the CMMI model which as we are reading it right now we believe this organization may not be doing?" Then ask, "Are we achieving the intent. If so, how?"

This may lead to an alternative practice, or a different "how-to" approach. Keep in mind that the CMMI focuses on "what" you must do, and you have many options related to "how" you do it. Agile approaches provide potential "how to" options [2]. Stress here the word "potential" because what can be a good "how to" in one organization may not be a good "how to" in another. Your "how to" options are not dictated by the CMMI model [1]. Those decisions should be made by your people based on your business needs.

Another good question to ask is, "Is there a problem in the organization because this practice does not seem to be done?" If the answer is no, then tell your TWG members to keep digging because they are likely to uncover a "local" practice. A "local" practice is something that is often not documented and is taken for granted in organizations, but is achieving the intent of a CMMI practice. An example of a "local" practice at BOND is something we referred to as "doorway" risk management. Risk management was ingrained in everything BOND did, which was part of the reason for their success. When a project leader had a risk he did not wait for a formal risk board meeting. He was in the "doorway" of his manager's office strategizing the risk mitigation immediately. We did not change this process, but we did document it and train it. Such "local" practices are common in many successful Agile organizations—and usually deserve more attention than they often get [3].

If, on the other hand, there is a problem in the organization, then the next discussion item for the TWG is to decide if they agree this organization needs to "stretch" by changing their behavior to resolve the problem right now. This is a very important discussion because you need to be sensitive to your organization's specific business needs and each time we agree to stretch it is critical that we know the problem we are solving with the stretch.

It is also critical to convey this rationale to the personnel in the organization that are affected so they understand why they are being asked to change their behavior. Behavior change is the hardest part of process improvement, but by providing solid rationale we can move the organization forward more rapidly and more effectively.

Discussing and agreeing to "stretch" areas and the rationale, and digging for "local" practices that we then document once found are the biggest differences in how we run an Agile TWG from a traditional TWG.

It is worth pointing out that an Agile approach was used to develop and roll out the processes at BOND incrementally and based on priority as identified in the gap analysis.

Added Value the CMMI Can Bring an Agile Organization

What has been described so far is how we go about documenting and deploying processes that are CMMI compliant in an Agile organization. But if your Agile organization is already successful, why go through all this effort?

The answer is because even successful Agile organizations have areas they need to improve and where behavior change is needed. We next describe a few examples from the BOND case study where behavior change was required, how we addressed this need, and how we achieved the buy-in from the software practitioners in the organization.

More About BOND

Successful organizations often start out as the brainchild of just a few individuals, and often those leaders keep a great deal of information inside their heads. This was the case at BOND. The leaders at BOND also took on a great deal of responsibility that would have typically been spread across many individuals in traditionally structured organizations. BOND's success led to rapid growth, which in turn led to a need to delegate. But this led to a problem.

The Delegation Problem at BOND

At BOND, in order to help maintain the successful Agile culture, it was decided to grow new leaders from the inside, rather than hire from outside. While this decision did help to maintain the desired Agile culture, it created a new problem. The new leaders were unsure of just what their new responsibilities entailed, and they were concerned because they were not being relieved of their previous responsibilities. This is an example of the type of critical information that came out from conducting the gap analysis with a focus on letting the people just talk about their job openly. This kind of information would not have been found by focusing on documentation alone, which is the common traditional approach to a gap analysis.

To address the delegation problem, as we extracted the management processes from the heads of the leaders at BOND and documented them, we also documented roles and responsibilities and were careful to keep both aligned. This led to tailored project lead training, which did not previously exist in the organization.

Tailored Project Lead Training

I emphasize here the word tailored because this training was not traditional project management training that you could purchase off-the-shelf. The focus of this training was on the "stretch" areas that the TWGs had agreed to. Keep in mind that if we agreed we could not see the value to change based on the TWG analysis it did not become a "stretch" and we did not do it. These cases did require some discussion during the formal appraisal, but since the rationale for decisions had been captured our lead appraiser understood and they caused no difficulty during our formal appraisal.

Why Focus on Stretches During Training?

People know how to behave the way they are behaving today in an organization. When new people come into an organization they learn quickly by observing what others are doing. This is not to say we ignored existing desired behavior in our training. We did highlight it, but we did not have to focus on it. On the other hand, the focus of our training was on "stretch" areas and related rationale because change takes time, and people respond best when they understand why they are being asked to behave differently.

Is Training With Focus on Stretches All That Was Required?

Changing behavior is the hardest part of process improvement. At BOND we used multiple approaches to address this challenge. First, because you cannot rely on people learning from their peers when you are trying to change an organization's current behavior, you do need training with rationale. However, training with rationale alone is insufficient because even when people understand the reason for change, when they return to their work environment human nature often leads them to first behave as they have been behaving in the past. As a result, at BOND we also instituted what we called "Sustainment" training which was short sessions often conducted as brown-bag lunch-time seminars where we provided reminder tips for areas we knew the organization was having trouble. We also instituted "coaching" to help with specific situations where people did not understand how to apply new expected practices.

How Did We Know Where BOND Needed Sustainment and Coaching Help?

In order to know where BOND personnel needed reminders during sustainment training, and additional coaching we need feedback mechanisms. These were provided through two sources; Product and Process Quality Assurance (PPQA) checks, and interactive workshops. When we initiated the process improvement program there were no independent quality checks happening in the organization. This is common in Agile organizations.

PPQA and Interactive Workshops

Some misunderstand the purpose of PPQA. A common myth is the belief that because "quality" is engineering's responsibility nothing else is needed. The purpose of PPQA is to provide "objective insight". There are multiple "how to" options to institute PPQA in an Agile organization. While some organizations use a "police force" approach, at BOND an approach was used where project personnel were rotated through the quality role providing more of a mentoring and sharing culture.

The training sessions where we focused on "stretch" areas were also conducted as interactive workshops. Besides being a time when practitioners learned about the company processes and the expectations with respect to stretch areas, they also became an opportunity for practitioners to share with each other issues and lessons.

The feedback from the PPQA audits and the interactive workshops was used to help focus lunch-time sustainment training sessions, and to improve processes and future training sessions.

The interactive workshops at BOND served multiple purposes. Typical Agile approaches do not address sharing across the organization, and training people in critical skill needs such as estimating, collaborating and handling sensitive issues, such as a difficult sub-contractor or customer. These were all topics that at times became a focus of the interactive workshops.

The Results

The purpose of this paper was to explain key techniques that were successfully employed to take a growing Agile organization to CMMI Level 3 while maintaining the organization's successful Agile culture. Feedback from both the leaders at BOND and their customers indicated noticeable improvement in cost and schedule management which can be attributed largely to the tailored project management processes and training focusing on the "stretch" areas. While there was concern at the start of the improvement project that the added effort required due to the CMMI would negatively impact team velocity, no noticeable impact was actually observed. In fact, the reverse was observed as on-time deliveries to customers actually improved. Surveys from developers taken during training workshops also indicated minimal impact was observed to their Agile approach (e.g. Scrum). When they were asked to behave differently they understood the rationale and why it was important to support the continued growth of the company.

It is also worth noting that the team felt the value to the organization was worth the minimal added effort and they felt the added tasks did not cause significant compromise or loss of Agile values. Key to achieving this result can be traced back to applying effectively the fundamental guidance in using the CMMI model described in the beginning of this paper. This guidance is critical to effectively integrating CMMI and Agile approaches in a way that does not cause the often-heard "non-value-added record-keeping" that too many organizations suffer from when implementing the CMMI the wrong way. Our success at BOND can also be attributed to three critical areas; the way we conducted the gap analysis, the way we ran the TWGs, and the way we achieved buy-in to needed changes.

With respect to the gap analysis keys to our success tied to our close attention to first gathering accurate data related to how people operated at the start of the effort, reporting clearly to the sponsors the potential specific patterns in the organization that needed to be addressed, and the approach used to handle potential weaknesses during the gap analysis.

With respect to the TWGs, the key to success was tailoring of the traditional TWG responsibilities to include training, requiring rationale for "stretches", and our approach to handling potential weaknesses in the TWG by digging for "local" practices when we could not uncover a related problem in the organization.

With respect to buy-in to needed changes keys to success tied to the focus of training on the rationale for "stretches", gaining feedback through a listening/mentoring PPQA culture, and follow up sustainment training and coaching in support of continual process improvement.

It is also worth noting that the main value of the CMMI effort from the customer perspective was more consistent product deliveries. Prior to the CMMI implementation, while customer satisfaction was generally good, there were specific cases of missed commitments due to the unexpected loss of key personnel, and the organization having no backup plan. Agile methods heavily rely on team members meeting their commitments. CMMI adds a focus on the organization providing improved support for trained resources that can be accessed across multiple projects in parallel, if necessary. This proved valuable at BOND as the organization grew and more projects needed to be managed in parallel.

The practical techniques described in this paper helped the BOND organization not only achieve a full CMMI Level 3 while maintaining their successful Agile culture, but also institute critical improvements the organization needed to help it continue to succeed as the organization continued to grow.[1]

Conclusion

This article focused on a single case study of a small growing Agile organization moving to CMMI Level 3. If you are facing similar challenges as the BOND organization and you are now wondering how to get started transitioning your organization to the CMMI, then you have missed a key point. As stated in the beginning of this paper, the CMMI is not a set of dictated practices. It is not something you should be "transitioning" your organization to. What you should do is start with a gap analysis and conduct it following the three keys outlined for conducting a gap analysis in an Agile organization. Then develop your process improvement plan with priorities established based on your gap analysis findings. Next get your TWGs going in the right direction by giving them the key rules for running TWGs in an Agile organization provided in this paper. If you use the CMMI as recommended you can effectively integrate the CMMI and Agile Development as BOND did gaining the benefits of the CMMI and maintaining your Agile values. In this brief article we cannot possibly answer all the questions you are likely to have. If you would like more detailed information on the BOND Case Study refer to [3].

For more information on how to integrate the CMMI and Agile Development in other situations, including organizations who are struggling to implement Agile approaches effectively and high maturity organizations seeking to increase their agility, refer to the additional case studies in the author's book [3].

Disclaimer:

CMMI® and CMM® are registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

ABOUT THE AUTHOR

Paul E. McMahon, Principal, PEM Systems, helps large and small organizations as they move toward increased agility and process maturity. He has taught Software Engineering at Binghamton University, conducted workshops on Engineering Process and Management, and published more than 40 articles. Paul is the author of two books, Integrating CMMI and Agile Development: Case Studies and Proven Techniques for Faster Performance Improvement and a book on collaborative development, Virtual Project Management: Software Solutions for Today and the Future. Paul is a certified Lean Six Sigma Black Belt and a Certified ScrumMaster.

Website: <http://www.pemsystems.com>
E-mail: pemcmahon@acm.org

REFERENCES

1. Chrissis, Konrad, Shrum, CMMI for Development: Guidelines for Process Integration and Product Improvement, Third Edition. Boston, Pearson Education, Inc., 2011.
2. Cockburn, Alistair, Agile Software Development: The Cooperative Game, Second Edition, Boston, Addison-Wesley, 2007.
3. McMahon, Paul E, Integrating CMMI and Agile Development: Case Studies and Proven Techniques for Faster Performance Improvement, Addison-Wesley, 2011

NOTES

1. For more information on the BOND Case Study refer to Chapters 4 and 5 in "Integrating CMMI and Agile Development: Case Studies and Proven Techniques for Faster Performance Improvement"

Extending the Range of Value of the CMMI To a New Normal

Don O'Neill, Independent Consultant

Abstract. Now that the CMMI® has been organized into three constellations for assuring an organization's capability to perform development, acquisition, and service, there is a need to extend the range of value of the CMMI to a new normal. As an organization improves its process maturity, strategic imperatives need to replace waste and neglect as the driver of CMMI value. Only those organizations able to elevate their game and transition from tactical to strategic use of the CMMI will be able to reap its full value.

While the traditional treatment of the value of the CMMI in terms of cost, schedule, productivity, quality, customer satisfaction, and ROI is sufficient to promote adoption of the CMMI and even to sustain a process improvement initiative through the early maturity levels, the value of the CMMI determined in this way is likely to be underestimated as the organization approaches higher maturity levels.

The value of the CMMI can be framed more strategically as the means for carrying out visionary statements of strategic intent in achieving measured outcomes in business and competitiveness, management and predictability, process and improvement, engineering and trustworthiness, and operations and dependability.

Not only that, the value of the CMMI to an organization varies depending on the domain of forces to which it must respond, such as, reputation, economics, mission, competitiveness, outsourcing, and high assurance. The penultimate value of the CMMI is the degree to which its ability to deliver satisfactory responses in a strategic context is demonstrated when faced with these competing forces. It is time to revisit why organizations should adopt the CMMI and to refresh the value proposition.

It is the responsibility and role of the change agent to unlock the value of the CMMI by strategically customizing the CMMI value to the organization. Change agents need to reach beyond compliance-oriented middle managers in composing more non-deterministic strategic statements of value in collaboration with senior executives in forging the new normal.

State of the CMMI

Watts Humphrey defined software process as the set of tools, methods, and practices used to produce a software product [1]. The quality of the software process largely determines the quality of the software products that result.

The CMMI is being adopted worldwide by government, military, and commercial organizations as the standard for process improvement. The CMMI is a framework of best practices that focus on assuring product quality through process performance (see Figure 1).

Prototyped in 1988 and now retired, the original CMM® focused on software processes [2]. Introduced in 2000, the CMMI focused on software development and was expanded to include systems engineering, product acquisition, integrated team, and requirements development. The CMMI is now organized into three constellations and has become the basis for assuring an organization's capability to perform software development (CMMI-DEV 2006), acquisition (CMMI-ACQ 2007), and service (CMMI-SVC 2009). The current CMMI is labeled Version 1.3 and was released in December 2010 [3].

Due to its origins, the CMMI lacks an explicit correlation to business alignment and strategic planning, sources of essential value to the enterprise [4]. In addition the CMMI may operate best in a closed system with top-down command and control decision-making [5]. In open organization environments with more diverse bottom-up consensus-based decision-making, other choices may be preferred. With pressure mounting on the value of the CMMI, the benefits of Agile and Interactive Development methods known since the 1970s [6], and the wide spread adoption of Six Sigma [7], the source and range of value of the CMMI are being questioned and tested. Even Watts Humphrey has expressed concern.

Asked about the direction the CMMI is headed, Watts Humphrey conceded that the CMMI has a problem with performance for high maturity organizations and specifically cited the use of process performance baselines and models by lead assessors [8]. He made a careful distinction between procedural (the what) and operational (the how) processes. Whereas, the procedural process depends on a bureaucracy to enforce it, the operational process depends on coaching a self-managing trusted workforce to apply its methods.

In accordance with the need to foster innovation, the bureaucratic top-down appraisal-driven compliance may be giving way to more diverse bottom-up self-directing team empowerment and self-determination. Just as the CMMI focuses on the what in assuring product quality through process performance, Agile deals with how to build software through well-defined methods that place an emphasis on increasing customer satisfaction. Similarly, Six Sigma further supplies the how with an emphasis on the systematic use of artifact templates, measurement, and control graphics in data-driven decision-making and the reduction of waste.

A New Understanding of the Value of CMMI

Change agents must now revisit their understanding of the value of the CMMI. The CMMI organization into three constellations spanning development, acquisition, and service and the

Maturity Level	Project Management	Engineering	Process Management	Support
Level 2	Project Planning (PP); Project Monitoring and Control (PMC); Supplier Agreement Management (SAM)	Requirements Management (REQM)		Configuration Management (CM); Process and Product Quality Assurance (PPQA); Measurement and Analysis (M&A)
Level 3	Integrated Project Management (IPM); Risk Management (RSKM)	Requirements Development (RD); Technical Solution (TS); Product Integration (PI) ; Verification (VER); Validation (VAL)	Organization Process Focus (OPF) Organization Process Definition (OPD) Organization Training (OT)	Decision Analysis and Resolution (DAR)
Level 4	Quantified Project Management (QPM)		Organization Process Performance (OPP)	
Level 5			Organization Innovation and Deployment (OID)	Causal Analysis and Resolution (CAR)

Figure 1. CMMI V1.3 Process Areas by Level and Category

expanded target audience of producers, buyers, and users of software products and systems bring with it change … change for the change agents as they take the lead in establishing a new normal of expectation for the value of the CMMI. It is time to revisit why organizations adopt the CMMI and to refresh the value proposition.

Change agents have systematically underestimated the value of the CMMI as they service the needs of middle managers seeking benefits that demonstrate compliance with the CMMI through tactical improvements, such as, cost, schedule, productivity, quality, customer satisfaction, and ROI. Instead change agents need to focus on the increasing value of software to the enterprise and engage senior executives by framing the value of the CMMI in their more strategic terms spanning innovative and visionary claims that enhance the reputation of the enterprise, promote superior economic achievement, meet mission performance expectation, achieve global competitiveness, promote trusted outsourcing, and demonstrate high assurance [9].

Framing the Value of the CMMI

Contrary to the arguments by some that the CMMI is unnecessary [10] and its value is overestimated, the real value of the CMMI is systematically underestimated.

1. In the small, the value of the CMMI is traditionally cast in terms of cost, schedule, productivity, quality, customer satisfaction, and return on ROI [11]. In accordance with the Theory of Expected Utility [12], these outcomes are thought to attain the most benefits and incur the least cost when using the CMMI.

2. Specifically, where the cost of quality includes both the cost to achieve quality and the cost of poor quality, defect avoidance and early defect detection are the principal drivers underlying these benefits [13]. The cost of quality, often consuming two-thirds of the engineering budget, is being cut in half through process improvement.

3. In addition, software productivity improvements approaching 50% have been experienced along with overall cost reductions of 25% [14].

4. While the use of these factors as markers of CMMI value may supply sufficient motivation to adopt the CMMI, especially an attractive ROI, the real value of the CMMI is likely to be underestimated.

The value of the CMMI can be viewed more comprehensively and is ultimately determined by the increasing value of software to the enterprise and the nation. This more expansive vision of software value must take into account the essential role of systems engineering and its tight coupling with software engineering.

1. In the large, the value of the CMMI lies in its role as an enabler of strategic software management. Strategic software management revolves around knowing what the customer needs most, aligning the best capability to provide it, understanding current practice, measuring its critical aspects, selecting the most promising changes, planning for lasting improvement, raising the ability to improve, and staying the course.

2. In framing the issue around strategic intent, means, and measured outcomes, the value of the CMMI can be leveraged in terms of strategic software management, and the statements of strategic intent can be cast directly in the context of the

Value of CMMI	Strategic Intent	Means	Measured Outcomes
Business	Competitiveness	Supplier Control Customer Control Competitor Control Threat Event Control	Staff Churn Personnel Turnover Open Requisitions Employee Moral Personnel Overtime Off-the-Clock Time Span of Responsibility Customer Loyalty Customer Satisfaction Release Frequency Time to Market Reuse Practice Open Source Innovation
Management	Predictability	Commitment Management Requirements Management Planning and Tracking Management Oversight Risk Management	Change Control Cost Control Schedule Control Earned Value Control Productivity Quality Control Span of Responsibility
Process	Improvement	Process Definition Measurement Training	Repeatability Predictability Control Schedule Control Capability Control Capacity Control
Engineering	Trustworthiness	Disciplined Software Engineering Completeness Correctness Consistency Rules of Construction Team Innovation	Reliability Availability Security Resiliency Traceability Defect Free Uniformity Complexity Control Usability Ideas generated, selected, and used
Operations	Dependability	Management Process Engineering Human Resources	Sustainability Repeatability Control Predictability Configuration Management Defect Management Span of Responsibility Capability Control Capacity Control

Table 1. Strategic Intent, Means, and Measured Outcomes

Industry Sector/ Elements of Value	Reputation	Economics	Mission	Competitiveness	Outsourcing	High Assurance
Telecommunications	•		•			•
Financial Services	•	•				•
Manufacturing		•		•	•	
Transportation			•			•
Medical	•		•			•
Utilities and Energy		•				
E-Commerce				•		
Defense			•			•

Table 2. Dominant Cultural Drivers by Industry Sector

business, management, process, engineering, and operations cultural drivers of the organization and its industry sector.

3. The adoption and expert use of the CMMI leverage the means through an organizational culture, professional environment, and process framework.

In reasoning about the value of the CMMI, the business value proposition revolves around how the issue of value is framed. As the means for carrying out statements of strategic intent and achieving measured outcomes, framing the value of the CMMI in the large focuses on the elements of strategic intent, means, and measured outcomes spanning business, management, process, engineering, and operations (see Table1):

1. Business and competitiveness [15] include control of suppliers, customers, competitors, and threat events [16] and their measured outcomes spanning staff churn, personnel turnover, open requisitions, employee morale, personnel overtime, off-the-clock time, span of responsibility, customer loyalty, customer satisfaction, release frequency, time to market, reuse practice, open source, and innovation.

2. Management and predictability include commitment management; requirements management; planning and tracking cost, schedule, and quality; configuration management; management oversight; and risk [17] management and their measured outcomes spanning change control, cost control, schedule control, earned value control, productivity, quality control, and span of responsibility.

3. Process and improvement include process definition, measurement, and training and their measured outcomes spanning repeatability, predictability control, schedule control, capability control, and capacity control.

4. Engineering and trustworthiness include disciplined software engineering; the standard of excellence for completeness, correctness, consistency, and rules of construction; and team innovation and their measured outcomes spanning reliability; availability; security; resiliency [18]; traceability; defect free; uniformity; complexity control; usability; and ideas generated, selected, and used.

5. Operations and dependability include sustainable management, repeatable and predictable process, trustworthy software engineering, and human resources capability and capacity both in-house and outsource and their measured outcomes spanning sustainability, repeatability, predictability control, configuration management, defect management, span of responsibility, capability control, and capacity control [19].

The Value of the CMMI Varies

The value of the CMMI varies in accordance with the forces that drive the organization. The culture of the organization is shaped by its strategically intended responses to these forces.

1. The industry sector in which an organization is a competing or participating member exerts influences associated with controlling suppliers, customers, competitors, and event threats. Some examples of industry sectors include telecommunications, financial, manufacturing, transportation, medical, utilities and energy, e-commerce, and defense.

2. The relative size, positioning, and longevity of an organization within its industry sector influence the mix of past, present, and future strategies and tactics it adopts. Some organizations find themselves anchored in the legacy of the past. Others simply glean the benefits of a prosperous economy without a plan for the future. Still others perhaps new on the scene, not well established, and without a legacy are banking on the future.

3. The software products and services and the mix of embedded, organic, and packaged offerings are driving forces in software production, fielding, and maintenance.

The value of the CMMI to an organization is different depending on the domain of forces to which it must respond. Where a valued aspect is dominant, such as, reputation and image, economics and finance, mission and continuity of operations, indicators of competitiveness, supply chain management and outsourcing, and trustworthiness and high assurance, an optimum response may result, thereby, simplifying the making of commitments, setting goals, and conducting tradeoffs. In less optimal situations, a blend of valued but competing aspects may lead to a more diverse response to these forces. Table 2 suggests the dominant cultural drivers by industry sector.

1. An organization driven by reputation and avoiding the risk of loss of trust may place a high value on trustworthiness and security along with the steps needed to assure these attributes. The telecommunications, financial services, and medical sectors where trust is all-important fit the reputation scenario.

2. An organization driven by economics may place a high value on profitability and attributes like cost control, productivity, and span of responsibility. The financial services, manufacturing, and utilities and energy sectors fit the economics scenario.

3. An organization driven by mission may place a high value on sustainability, capability control, and capacity control as well as reliability, availability, security, and resiliency. The telecommunications, transportation, medical, and defense sectors fit the mission scenario.

4. An organization driven by competitiveness may place a high value on release frequency, time to market, and innovation as well as cost and schedule control and predictability control. The manufacturing and e-commerce sectors fit the competitiveness scenario.

5. An organization driven by outsourcing may place a high value on release frequency, time to market, and innovation as well as quality control, configuration management, and span of responsibility of onshore staff. The manufacturing sector fits the outsourcing scenario.

6. An organization driven by high assurance may place a high value on trustworthiness including quality control, defect free, predictability control, resiliency, and frequency of release. The telecommunications, financial services, transportation, medical, and defense sectors fit the high assurance scenario.

Table 3. Ranking Cultural Drivers and CMMI Categories

Cultural Drivers/ CMMI Categories	Reputation	Economics	Mission	Competitiveness	Outsourcing	High Assurance
Project Management	2	1	3	1	2	3
Product Engineering	1	2	1	2	3	2
Process Management	3	3	2	3	1	1

Table 4. Ranking CMMI Categories, Cultural Drivers, and Leading Measured Outcomes

Cultural Drivers/ CMMI Categories	Reputation	Economics	Mission	Competitiveness	Outsourcing	High Assurance
Project Management	2 Release Frequency	1 Span of Responsibility	3 Quality Control	1 Time to Market	2 Change Control	3 Quality Control
Product Engineering	1 Defect Free	2 Complexity Control	1 Resiliency	2 Innovation	3 Traceability	2 Resiliency
Process Management	3 Schedule Control	3 Capability Control	2 Repeatability	3 Capacity Control	1 Predictability Control	1 Predictability Control

Table 5. Ranking CMMI Constellations, Cultural Drivers, and Leading Measured Outcomes

CMMI Constellation/s Cultural Drivers	Reputation	Economics	Mission	Competitiveness	Outsourcing	High Assurance
Development	1 Defect Free	3 Complexity	3 Quality Control	2 Innovation	2 Traceability	1 Quality Control
Acquisition	3 Schedule	1 Span of Responsibility	2 Repeatability	3 Time to Market	3 Predictability Control	3 Predictability Control
Service	2 Release Frequency	2 Capability Control	1 Resiliency	1 Capacity Control	1 Change Control	2 Resiliency

Table 6. Description of Leading Measured Outcomes

Outcomes	Description
Capability Control	Managing and sustaining the knowledge, skills, and abilities of enterprise and project personnel to perform the standard organization process definition and its project tailoring.
Capacity Control	Managing and sustaining the personnel workforce with the knowledge, skills, and abilities of enterprise and project personnel needed to perform the standard organization process definition and its project tailoring.
Change Control	Managing changes to a baseline to form a new baseline.
Complexity Control	Maintaining intellectual control over the interfaces, dependencies, and interactions among software components within a system.
Defect Free	Absence of errors, faults, and failures.
Innovation	The intersection of invention and insight leading to the creation of something of value.
Predictability Control	The application of statistical process control to cost, schedule, and quality metrics and the control of the resulting variances.
Quality Control	Managing quality expectation and actual quality performance.
Release Frequency	Duration between the issuance of quality assured product updates to the field.
Repeatability	The degree to which a process description is faithfully carried out on successive applications.
Resiliency	The ability of a system of systems to anticipate, avoid, minimize, withstand, and recover from the affects of adversity, whether manmade or natural, under all circumstances of use.
Schedule Control	Managing schedule estimation, budgeting, change orders, and actual schedule performance.
Span of Responsibility	Total number of source lines of code on the project divided by the total head count on the project.
Time to Market	Duration between the time of conception and the ship date of a product or service.
Traceability	The alignment of software life cycle artifacts.

Achieving the value of the CMMI in actual application in the wild varies with the profile of the project and organization. The organizational challenges in culture, governance, shared ownership, and accountability may be larger than the challenges of information technology and software engineering [20]. Table 3 ranks the cultural drivers and CMMI categories of project management, product engineering, and process management. Table 4 shows these rankings along with the leading measured outcomes. Table 5 shows these rankings arranged by CMMI constellation. See Table 6 for a description of leading measured outcomes.

Particular CMMI Process Areas are associated with leading measured outcomes. See Table 7 for CMMI Process Areas by Cultural Drivers Leading Measured Outcomes.

Table 7. CMMI Process Areas by Cultural Drivers and Leading Measured Outcomes

Cultural Drivers/ Measured Outcomes	Reputation	Economics	Mission	Competitiveness	Outsourcing	High Assurance
Defect Free	REQM, M&A, PPQA OPD, OT, IPM, TS, PI, VER, VAL QPM, OPP					
Span of Responsibility		PP, PMC, M&A OPD, OT, IPM				
Resiliency			RD, TS, RSKM			RD, TS, RSKM
Time to Market				REQM, PP, PMC		
Predictability Control					PP, PMC, M&A, PPQA OPD, OT, IPM	PP, PMC, M&A, PPQA OPD, OT, IPM
Release Frequency	REQM, PP, PMC					
Complexity Control		REQM, CM RD, TS OPM, OPP				
Repeatability			OPPD, OT			
Innovation				OID		
Change Control					REQM, CM	
Schedule Control	PP, PMC					
Capability Control		OPD, OT, IPM				
Quality Control			PPQA			PPQA
Capacity Control				OPD, OT, IPM		
Traceability					REQM, CM RD, TS, VER, VAL	

Conclusion

While the value of the CMMI determined in the traditional way is sufficient to promote adoption of the CMMI, the value of the CMMI determined more strategically in terms of the means for carrying out statements of strategic intent in achieving measured outcomes in business and competitiveness, management and predictability, process and improvement, engineering and trustworthiness, and operations and dependability reveals the real value. When the industry sector forces and their cultural drivers, such as, reputation, economics, mission, competitiveness, outsourcing, and high assurance are taken into account, a deeper understanding of which CMMI categories and process areas need to be emphasized is the result.

1. For the enterprise considering adopting the CMMI as its framework for process improvement, framing the value of the CMMI in terms of cost, schedule, productivity, quality, customer satisfaction, and ROI is recommended. Here it needs to be understood that the CMMI may operate best in a closed system with top-down command and control decision making and that there is a growing preference for open organization environments with more diverse bottom-up consensus-based decision making where other choices may be preferred.

2. For the enterprise already engaged with the CMMI but seeking to extract the true value of the CMMI in the context of industry sector forces and intent on maximizing that value in terms of cultural drivers and specific strategic intents, framing the value of the CMMI more strategically in terms of measured outcomes in business and competitiveness, management and predictability, process and improvement, engineering and trustworthiness, and operations and dependability is recommended. Here it needs to be understood that the CMMI lacks an explicit correlation to business alignment and strategic planning and that innovative strategic thinking is required to connect the CMMI with these sources of essential value to the enterprise.

Disclaimer:

CMMI® and CMM® are registered in the U.S. Patent and Trademark Office by Carnegie Mellon University

ABOUT THE AUTHOR

Don O'Neill is a seasoned software engineering manager and technologist currently serving as an independent consultant. Following his 27 year career with IBM's Federal Systems Division, Mr. O'Neill completed a three-year residency at Carnegie Mellon University's SEI under IBM's Technical Academic Career Program and has served as an SEI Visiting Scientist.

In his IBM career, Mr. O'Neill completed assignments in management, technical performance, and marketing in a broad range of applications including space systems, submarine systems, military command and control systems, communications systems, and management decision support systems. He was awarded IBM's Outstanding Contribution Award three times:

1. Software Development Manager for the GPS Ground Segment.
2. Manager of the FSD Software Engineering Department responsible for the origination of division software engineering strategies, the preparation of software management and engineering practices, and the coordination of these practices throughout the division's software practitioners and managers.
3. Manager of Data Processing for the Trident Submarine Command and Control System Engineering and Integration Project responsible for architecture selections and software development planning.

Mr. O'Neill served on the Executive Board of the IEEE Software Engineering Technical Committee and as a Distinguished Visitor of the IEEE. He is a founding member of the Washington DC Software Process Improvement Network and the National Software Council and served as the President of the Center for National Software Studies from 2005 to 2008. He was a contributing author of "Software 2015: A National Software Strategy to Ensure U.S. Security and Competitiveness", a report on the Second National Software Summit. Mr. O'Neill has served as a reviewer of National Science Foundation software engineering research proposals and has served as a member of the NIST Software Assurance Metrics and Tool Evaluation Advisory Committee (2006-2008). He has authored Business Case articles for the CERT Build Security In web site. His current research is directed at public policy strategies for deploying resiliency in the nation's critical infrastructure.

Mr. O'Neill is an active speaker on software engineering topics and has numerous publications to his credit. He has a Bachelor of Science degree in mathematics from Dickinson College in Carlisle, Pennsylvania.

REFERENCES

1. Humphrey, Watts S., "Managing the Software Process", Addison-Wesley Publishing Company, Inc., 1989, 494 pages, ISBN 0-201-18095-2
2. Paulk, Mark C., "The Capability Maturity Model: Guidelines for Improving the Software Process", Addison-Wesley Publishing Company, 1995, 441 pages, ISBN 0-201-54664-7
3. Phillips, Mike and Sandy Schrum, "Process Improvement for All: What to Expect from CMM Version 1.3", CrossTalk: The Journal of Defense Software Engineering, Vol. 23 No. 1, January/February 2010, Hill AFB, Salt lake City, Utah
4. Heston, Keith M. and William Phifer, "The Multiple Quality Models Paradox: How Much 'Best Practice' is Just Enough?", SEPG North America 2010, Match 2010, Savannah, Georgia
5. Constantine, L.L., "Constantine on Peopleware", Yourdon Press, Englewood Cliffs, N.J., 1995
6. Larman, Craig, "Agile & Iterative Development: A Manager's Guide", Pearson Education, Inc., 2004, ISBN 0-13-111155-8, pages 82-85
7. Sirvastave, Nidhi and Sathya Murthy, "Harvesting CMMI Benefits- The Six Sigma Sickle, SEPG Conference, 2006
8. Humphrey, Watts, "An Interview with Watts S. Humphrey", CrossTalk: The Journal of Defense Software Engineering, Vol. 23 No. 4, September/October 2010, Hill AFB, Salt lake City, Utah
9. Moss, Michele, Kristy Mosteller, Stephanie Shankles, "Assurance for CMMI", SEPG North America 2010, March 2010, Savannah, Georgia
10. Agile CMMI blog, December 2007, <http://www.agilecmmi.com/2007/12/cmmi-is-overrated-and-unnecessary.html>
11. "Performance Results of CMMI-Based Process Improvement", CMU/SEI-2006-TR-004, August 2006, SEI Technical Report
12. Poulton, E.C., "Behavioral Decision Theory: A New Approach", Cambridge University Press, 1994, 334 pages
13. Reitzif, Rolf et al, "Calculating CMMI-Based ROI", 19th Annual SEI Software Engineering Process Conference, March 26-29, 2007, Austin, TX
14. McLoone, Peter J. and Sharon L. Rohde, "Performance Outcomes of CMMI -Based Process Improvement", DACS SoftwareTech News, March 2007
15. Competitiveness: The ability of products and services to withstand the test of international markets while maintaining or boosting the real wages of the workers who produce them. (Council on Competitiveness, Washington, D.C.)
16. O'Neill, Don, "Introducing Global Software Competitiveness", CrossTalk, The Journal of Defense Software Engineering, Vol. 16 No. 10, October 2003, Hill AFB, Salt lake City, Utah
17. Risk: Uncertainty and the prospect for loss or gain depending on the outcome of an event.
18. Resiliency: The ability to anticipate, avoid, withstand, mitigate, recover from the effects of adversity whether natural or manmade under all circumstances of use.
19. A software system that is trustworthy sustains attributes associated with properties, such as, completeness, correctness, consistency, predictability, availability, dependability, interoperability, security, safety, resilience, privacy, and usability and does so under all circumstances of use.
20. Prieto, Daniel B. and Dr. Steven Bucci, "Meeting the Cyber Security Challenge: Empowering Stakeholders and Ensuring Coordination", IBM U.S. Federal White Paper, 2010

DoD Agile Adoption:

Necessary Considerations, Concerns, and Changes

Mary Ann Lapham, Software Engineering Institute, Carnegie Mellon University

Abstract. Today's DoD acquisition environment relies on the DoD 5000 series of guidelines. Nothing in the DoD 5000 series precludes the use of Agile methods. In fact, Agile methods provide both tactical and strategic benefits. However, achieving these benefits is not likely to occur without changes to the traditional DoD mindset.

Introduction

This article summarizes the SEI Acquisition Support Program's exploration of using Agile approaches in software-intensive systems developed or being developed in the DoD. Our work to date has been to provide prudent, pragmatic advocacy of Agile methods for those within DoD who want or have to implement those methods. We have identified issues and challenges to overcome when implementing Agile in a DoD environment. These issues and challenges are summarized herein.

For purposes of this article, Agile is defined as: Agile: An iterative and incremental (evolutionary) approach to software development which is performed in a highly collaborative manner by self-organizing teams within an effective governance framework with "just enough" ceremony that produces high-quality software in a cost effective and timely manner which meets the changing needs of its stakeholders.[1]

Further, the terms "Agile methods" or "Agile approaches" are commonly used throughout to characterize a set of disciplined incremental methods that involve strong, continuous end-user collaboration, frequent (two to four week) work in progress deliveries, and techniques such as continuous integration and test-driven development. Although all Agile methods are incremental, not all incremental methods reflect Agile properties.

Since the SEI work began, there has been considerable movement within the government and DoD to identify and implement a new acquisition process that can take advantage of Agile methods. Attachment 2 of the "804 report" [1] provides Interim Acquisition Guidance for Defense Business Systems.

Our review of the DoD 5000 series showed that there are no interpretations that directly preclude or limit the use of Agile methods within the DoD. There are some constraints, challenges, and even some supportive instances within the policy and instruction. Agile methods, "Can provide both tactical and strategic benefits. The tactical benefits of lower cost within schedule and increasing quality are important; however, the strategic benefits of being responsive and being able to adjust to the current situation more rapidly might be of even greater value [2]. This could be a huge factor in today's world, where the DoD needs to get results faster and be better aligned with changing needs" [3].

Policies, regulations and other governing documents aside, there are underlying concerns that will form the basis for adopting Agile within the DoD. The main difference between using Agile and a more traditional method is the requirement for different management and technical approaches if the advantages of Agile are to be fully realized. In addition, the Program Management Office (PMO) needs to determine how proficient it will be at organizational change [4].

Potential Barriers and/or Differences From Traditional Methods

Interviews with several DoD programs that are using or have used Agile methods combined with a review of relevant literature revealed some of the areas where barriers and/or differences from traditional methods are encountered [3]:

- Acquisition lifecycle: Some lifecycle phases lend themselves to the use of Agile better than others. Remember to consider Agile processes and so that contractually binding documents, such as the request for proposals, and statement of work, support those processes and practices. One particular stumbling block for the adoption of Agile tends to be capstone technical review events such as preliminary design review and critical design review. Agile methods typically do not produce the types of documentation expected at these milestones. Instead, they provide working prototypes and, in some cases, a subset of requirements implemented as usable software. Therefore, expectations and criteria for acceptance need to be established at the beginning of the contract that meet both the contractual needs and allow for the use of Agile methods. Since Agile produces the final product iteratively, the expectations and criteria for acceptance need to be compatible.
- Team environment: A central concept to Agile is the small, dynamic, high-performing cross-functional team (or teams depending on the size of the program). Testing is done concurrently within the team with continuous integration [5]. The teams expect input from the end users throughout this process. Each team usually conducts regular reflection and adaption called retrospectives. The government team needs to understand and support this way of doing business. Otherwise, using Agile will have less than optimal results.
- End-user access and involvement: One of the key tenets stated in the Agile Manifesto, the document that, since 2000, has guided adopters of Agile approaches, is "Customer Collaboration over Contract Negotiation."[2] This is usually accomplished by having continuous contact with the end

user. In many instances, the end user is an integral member of the iteration team. This is not always practical in the DoD environment, especially with joint programs and the myriad of stakeholders DoD software-reliant systems serve. In addition, the real end user is an operational person who may not have any experience in the acquisition career field while the acquirer may or may not have operational experience. The contractor and government usually solve this problem by agreeing on a proxy for the end users' day-to-day interaction and inviting end users to all demos. This end user interaction is important in successful projects using Agile [6].

• Training and coaching to provide knowledge of Agile: Many of the Agile concepts are not new, but the subtleties and nuances of each Agile method can be new to the uninformed. To overcome this, all PMO staff should be trained in the contractor's method of choice [3]. It is important to set aside funding for initial and ongoing training and support. Without the requisite training, misunderstandings will certainly occur and could have disastrous consequences. A coach and/or an Agile advocate who has "clout" within the PMO is a good addition to the PMO staff. Their presence can answer daily questions, help resolve issues before they become problems and help to ensure the program runs smoothly from an Agile perspective. The Agile advocate/ coach must have authority; otherwise they will get lost in the chorus of voices demanding to be heard.

• Oversight including milestone reviews, documentation, and evaluation (metrics): Traditionally, the government uses milestone reviews, documentation, and evaluation metrics to monitor and evaluate contractor progress on and/or review specific aspects of the proposed technical software solution [7]. Typically, the expectations and criteria for milestone reviews and documentation are negotiated at contract award and certainly well before the milestone event occurs [8]. This practice is not different for programs using Agile methods. However, documentation for an Agile program is just enough to meet the minimal set of technical and programmatic needs and provide continuity for the team. This type of documentation is not usually enough for capstone events. Thus, the negotiations need to determine what is acceptable for the program and yet will work within the Agile environment. Tailoring typically takes on additional importance. Some keys that are useful in assuring that the ultimate outcome is achieved:

* Confirm all parties have a stake in the outcome or as the Defense Science Board has stated have some "skin in the game" [9].

* Determine how regulatory documentation that does not necessarily contribute directly to development activities will be created.

* Agree to the intent and content of each artifact.

* Make sure all requirements levied by guiding instructions, directives, etc are expressly met.

One analogy for oversight within the Agile community could be what the military calls "Commander's intent." Commander's intent provides a clear, concise, and focused statement of intent. Thus, the mission can continue, even if the operation does not go as planned [10]. For Agile, the overall plan is the intent. If the plan does not work as expected, the development team alters the plan with the intent in mind. This requires trust, collaboration and relationship building, which are core ideas for Agile. Performing Agile implementations requires that the oversight method, documentation, and form of metrics be thoroughly negotiated and agreed upon in advance of starting the program. When doing this negotiation, keep in mind that less formal does not mean undisciplined. Agile programs tend to be less formal, but highly disciplined.

• Rewards and incentives: Rewards and incentives for Agile teams focus on the team. This seems to be contrary to the traditional individual based reward system in place on most programs where the "hero" gets the award. Unless the government is doing internal development, the majority of change in this reward model is left to the contractor. However, the government can assist by considering incentives that embrace and foster change and sharing of data. "Personnel need to be incented to do significant adoption of planning and strategy for the technology shift and related business, legal, and operational aspects" [3].

• Team composition: The team composition for Agile developers is different than on traditional teams. Thus, the government should consider that their team will also have a different composition. Two important positions that are new to most government teams are those of Agile advocate and end-user representative. An Agile advocate, as described in Training and coaching above, provides real-time answers to immediate Agile issues for the government team. The end-user representative not only needs to represent the end users, but must have the authority (within delegated limits) to direct the contractor. Without skills in modern software development approaches, the government program office may have issues with oversight, which are quickly visible in the fast paced Agile world.

• Culture: Culture is the customary knowledge, beliefs, behavior, and traits displayed by an acquisition organization or contractor [3]. A brief comparison of some typical cultural elements is shown in Table 1. The same elements can have significantly different instantiations depending on the method employed [8].

"Traditional project managers focus on following the plan with minimal change but the Agile manager focuses on adapting successfully to inevitable change" [4].

This illustrates two very different mindsets. If the government is serious about adapting Agile methods, then they will have to modify their mindset so that they view software lifecycles from other perspectives than the traditional metaphor [11]. This will not be easy and does not mean traditional methods should be totally abandoned. The culture change needs to provide flexibility so that traditional and Agile methods can be employed when and where needed. Neither method provides a solution to all problems.

For example, one possible action that could be taken to bring change to the rewards system is to make some or all rewards team based. Rewards can be other than monetary, such as choice of assignment, mentoring, training, etc. Downplaying merit increases and associating career accomplishments and

Element	Agile DoD	Traditional DoD
Organizational Structure	• Flexible and adaptive structures; • Self organizing teams, • Co located teams or strong communication mechanisms when teams are distributed	• Command and control structures that are difficult to change • Hierarchical, command and control-based teams
Rewards System	• Team is focus of rewards • Sometimes team itself recognizes individuals	• Individual is focus of the reward system
Communications & Decision Making	• Daily stand up meetings, • Frequent retrospectives, • Information radiators[5] to communicate critical project information; • Evocative documents to feed conversation; • "Just enough" documentation. • Control and discipline comes from the Agile team itself.	• Top down communication; External regulations, policies and procedures tend to drive the work. Activities and processes documented; • Traditional, representational documents used by the PMO throughout the development life cycle to oversee the progress and discipline of the developer through formal and informal reviews.
Staffing Model	• Cross functional teams including all roles across the life cycle throughout the lifespan of the project; • Agile advocate or coach • End-user representative	• Uses traditional waterfall model with separate teams, particularly for development and testing • Different roles (e.g. developer, tester) are active at different defined points in the life cycle and are not substantively involved except at those times

Table 1. Comparison of Some Agile and Traditional DoD Cultural Elements

milestones with promotions is one strategy. Another strategy is to let the team naturally recognize its heroes and include an appreciation step during your retrospective [8].

A final word about culture. There is a big difference between doing Agile and being Agile. Picking an Agile process and following it step by step without fully embracing the culture can provide some benefit. However, if being Agile is the goal, then a culture of agility needs to be created [12]. The culture goes beyond using an Agile software delivery process, it seeks to change what the team values, measures, and delivers (i.e., placing value on collaboration and personal interactions, working software and adjustment to change) [8].

• Integration and test: Continuous integration and test of some form is done within Agile teams. This is contrary to the traditional approach where integration is done at the end of a release cycle. If final integration and test is being used for system acceptance, then most likely an independent external team will conduct the work. However, the continuous integration and test during the development using Agile methods should mean that there are less risks to be overcome as more issues will have been found earlier in the lifecycle. Additionally, there should be less risk of user rejection since testing by the Agile teams puts validation before verification through the involvement of the user.

• Managing Agile programs: The Agile approach to project execution places demands upon all personnel that are still traditional but it also differs from other execution environments. The managerial role is uniquely affected by the features of the Agile approach. Both the acquiring-side and execution-side[3] managers become leaders, coaches, expeditors, and champions.[4] As a leader, the executing manager needs to spend more time with the team to help create a "trust factor" so that delegating important tasks can easily be accomplished. The acquiring manager needs to determine who to designate as the on-site representative to maintain adequate visibility into the fast emerging product.

As a coach, both managers need to assist their personnel in making the transition to the fast tempo, high interaction environment that typifies Agile projects. This is often accomplished by including someone who has the role of Agile coach for the project. As an expeditor, the executing manager

needs to identify and quickly remove any organizational and operational impediments. The acquiring manager needs to secure appropriate status information without unduly interfering with the tempo of Agile development using negotiation and establishing trust with the executing manager. As a champion, the executing manager will need to translate the unfamiliar, if not foreign, Agile model for the upper-level management and other managerial stakeholders. In addition to this, the acquisition manager will have to maintain buy-in by external funders and stakeholders. This will include providing a portrayal of project status and accomplishments that is accurate as well as bridging the cultural gap that exists.

Road to Agile Adoption

During our interviews, the two main reasons within the DoD for moving to Agile are a burning platform (i.e., if the program does not change its current development practice to improve outcomes, it is likely to get cancelled); and urgency of delivery, i.e., an operational need that cannot wait for traditional delivery times is mission-critical enough to warrant a different acquisition approach [8].

We also found a third, perhaps more compelling reason to move to Agile methods. Section 804 of the National Defense Authorization Act for Fiscal Year 2010 specifies that information technology systems, "be designed to include (A) early and continual involvement of the user; (B) multiple rapidly executed increments or releases of capability; (C) early, successive prototyping to support an evolutionary approach; and D) a modular open-systems approach" [1]. The fact that Agile methods are more compatible "out of the box" with all four of these directives than typical IT acquisition practices is an encouraging sign that appropriate use of these methods in the future will be supported.

For those who have been using Agile methods for some time, some common themes that characterized continuing motivation for change included:

- A sense of true accomplishment when they delivered a release that they knew incorporated functionality the end user needed.
- A short time span for seeing the differences their work made to their end users.
- Encouraging (often laudatory) user feedback that clearly communicated the value of their approach.
- Consistent ability to meet or exceed user expectations.
- Previous inability to deliver value within agreed timespans and costs.

In order to adopt Agile methods, best practices in adoption and organizational change management need to be considered. Some of these topics are:

- Understanding your adopter population: [13] By this we mean understand the characteristics of the people both as individuals and as a group. For those in the DoD who have adopted Agile methods, they have been pathfinders in terms of finding ways to "work Agile" in an environment that demands artifacts and evidence based on "working traditional." Successful adoption across a wide spectrum of appropriate DoD programs will not occur until more communication and implementation support mechanisms are available [14].
- Understanding the cycle of change: Change takes effort and time [15]. From our interviews, it was common to phase adoption of Agile methods over a period of time to allow the staff to get accustomed to a new set of practices.
- Understanding your adoption risks: Know where you are in terms of practices, skills, sponsorship, and values. The adoption approach used by the majority of programs interviewed heavily leveraged external training and coaching [16].
- Building transition mechanisms to mitigate adoption risks: Some potential mechanisms are articles in CrossTalk, Defense Acquisition News, etc. on programs successfully using Agile methods and conference tracks and workshops that highlight the benefits and risks associated with adopting Agile practices [17].

Conclusion

Agile methods can provide the benefits of being responsive and being able to adjust to the current situation faster than when using traditional methods. Adopting Agile methods is not without work to overcome barriers. Others have done so and there is a wealth of information starting to accumulate to assist organizations wanting to make this change. The authors of the two papers summarized here are continuing to research this arena and add to the body of knowledge available for DoD use.

Disclaimer:

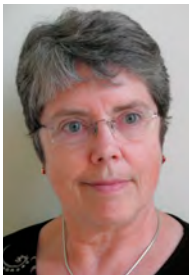
REFERENCES

1. Office of the Secretary of Defense, "A New Approach for Delivering Information Technology Capabilities in the Department of Defense", Report to Congress, Pursuant to Section 804 of the National Defense Authorization Act for Fiscal Year 2010, November 2010.
2. Boehm, Barry and Turner, Richard – Balancing Agility and Discipline – A Guide for the Perplexed. Addison-Wesley, 2004.
3. Lapham, Mary Ann, et al. "Considerations for Using Agile in DoD Acquisition." CMU/SEI-2010-TN-002, April 2010.
4. Highsmith, J., "Advanced Agile Project Management Seminar," in Agile Development Practices Conference, Orlando, FL, 2009.
5. Leffingwell, D., Scaling Software Agility: Best Practices for Large Enterprises. Addison-Wesley, 2008.
6. The DoD Agile Innovations Conference, The Association for Enterprise Integration (AFEI), Washington, D.C., December 2010.
7. Defense Acquisition Guidebook, Dated August 5, 2010, Defense Acquisition University, August 11, 2010,
8. Lapham, Mary Ann, et al. "Agile Methods – Detailed Exploration into Selected DoD Management and Acquisition Concerns." CMU/SEI-2011-TN-002, publication expected June 2011.
9. Defense Science Board, Report of the Defense Science Board Task Force on Department of Defense Policies and Procedures for the Acquisition of Information Technology, Office of the Under Secretary of Defense for Acquisition, Technology and Logistics, Washington, D.C., March 2009.
10. <http://www.globalsecurity.org/military/library/report/call/call_98-24_ch1.htm>
11. Nerur, S., Mahapatra, R., & Mangalaraj, G. (2005). Challenges of Migrating to Agile Methodologies. Communications of the ACM, 48(5), 73-78. doi: 10.1145/1060710.1060712
12. Sidky, A. and Smith, G., Becoming Agile in as Imperfect World, Manning Publication Co., 2009.
13. Rogers, E., Diffusion of Innovations 5th Edition, Free Press a Division of Simon & Shuster, Inc., 2003.
14. Moore, G., Crossing the Chasm, HarperBusiness, 1991.
15. Weinburg, G., Quality Software Management, Vol 4., Anticipating Change, Dorset House Publishing, 1997.
16. Garcia, S. and Turner, R., CMMI Survival Guide: Just Enough Process Improvement, Addisdon-Wesley, 2006.
17. Conner, D.R., and Patterson, R.W., "Building Commitment to Organizational Change," Training and Development Journal (April 1983): 18-30.

NOTES

1. <http://www.agilemodeling.com/essays/agileSoftwareDevelopment.htm>
2. See <http://Agilemanifesto.org/history.html>
3. The executing-side manager could be a development contractor or part of an organic government team, such as an Air Logistics Center team
4. The common traits takes inspiration from Dean Leffingwell [5] then alters and expands them to address inserting Agile practices into DoD acquisition.
5. Information radiator – is a large, highly visible display used by software development teams to track progress. The term was first coined by Alistar Cockburn. See <http://www.atlassian.com/wallboards/information-radiators.jsp>

ABOUT THE AUTHOR



Mary Ann Lapham, PMP, is a senior member of the technical staff at SEI at Carnegie Mellon University. She is currently working in the Acquisition Support Program's US Air Force team supporting various Air Force programs to improve acquisition practices in acquiring software intensive systems. In addition, Ms. Lapham has participated/led Independent Technical Assessments for DoD and non-DoD clients and is an instructor for the Software Acquisition Survival Skills. Her current research is focused on the use of Agile methodologies in various DoD contexts. Her B.A. and M.A. in Mathematics are from the State University of New York.

Mary Ann Lapham
4500 Fifth Avenue
Pittsburgh, PA 15213
Office: 412.26.5498
Fax: 412.268.5758
E-mail: mlapham@sei.cmu.edu

FOR CONFERENCE & TRADE SHOW INFORMATION, VISIT WWW.SSTC-ONLINE.ORG

24th Annual

SSTC
Systems & Software Technology Conference

REGISTER TODAY!

ACQUISITION
- Sustainment Cost Estimation
- Cyber Acquisition
- Measuring Enterprise Performance
- Reuse of Complex Systems
- Mapping & Modeling
- CMMI, ISO, Baldridge
- Life Cycle Cost Estimation

AGILE DEVELOPMENT
- Agile Systems Engineering
- Agile & Architecture
- Extreme Agile
- Surveillance Points
- Agile Modeling & Simulation
- Agile & Product Quality

ARCHITECTURE
- Net-Centricity
- SOAs in Embedded Systems
- Mobile Access to Enterprise Data
- Enterprise Architectures

CLOUD COMPUTING
- Migration Strategies
- Cost Considerations
- Mobile options
- Security Considerations

CYBER SECURITY
- Trustworthy Software
- Information Assurance
- Cyber Hardening
- Web & Mobile Security
- Securing Android Technology

23-26 APRIL 2012
MARRIOTT DOWNTOWN HOTEL
SALT LAKE CITY, UTAH

WAR FIGHTING TECHNOLOGIES

ENHANCE ADVANCE MODERNIZE

Increasing Workforce Capabilities – Solving Tomorrow's Challenges

Physically fit, totally equipped, and completely ready to defend the interests of the United States of America. That is what we have come to expect and appreciate from today's warfighter. We, who provide those men and women with the systems and software they need to do their job, must be similarly fit, equipped, and ready to perform our responsibilities. That isn't just hyperbole. It is a reality that every systems and software professional who supports the Department of Defense must embrace.

The challenges facing today's systems and software professionals are no less daunting than in years past:

- How can we maintain system security in an increasingly mobile environment?
- How do we provide secure global reach back to enterprise data?
- How do we develop more accurate, robust, and thus more reliable sustainment and/or lifecycle cost estimates?
- Why do we continue to underestimate the total ownership costs?
- How can we reduce or minimize the number of requirements changes, post system deployment?
- How can we close the gap between the system, as built to requirements specifications and the system, as expected by the end-users?

How do we answer the myriad of questions surrounding migration to 'the cloud'? Should we develop a private cloud capability? Should we fully embrace 'the cloud' and transition both applications and data? Is there a hybrid (part private – part public) solution that meets our needs? If we employ a non-private cloud solution, how do we maintain information security? … etc. etc. etc.

That is only a small, partial list of the issues facing today's systems and software communities. In today's extreme DoD fiscal environment, local, point, or stove-piped solutions cannot be enabled by encouraging everyone to hunker down in individual foxholes to search for solutions. Collaboration across domain, organizational, and command boundaries is also a reality that must be embraced.

SSTC2012 has been planned, designed, and is being executed to achieve two fundamental goals:

1) Provide a knowledge-rich conference that enables each attendee to return to their workplace more knowledgeable, with a broader perspective on the challenges and choices they face, and with an increased desire and commitment to doing more without more.

2) Provide senior military leaders the opportunity to provide guidance and share their perspectives with the systems and software communities.

Enabled and emboldened by the commitment of the Software Maintenance Group at Hill AFB (309th SMXG) and Utah State University, the Software Technology Support Center (STSC) is committed to revitalizing and restoring the SSTC to its previous position as the 'must attend' conference supporting DoD software and systems professionals.

The graphic on the facing page highlights the spectrum of topics that will be addressed during SSTC2012. You'll notice an amazing blend of leading edge topics like Security Considerations for Cloud Computing alongside innovations and improvements being made to more traditional topics like Life Cycle Cost Estimation. From issues of cyber security in android technologies to applying the principles of agile development to the systems engineering process, SSTC2012 will inspire, enable, and encourage software and systems professionals to think about the choices and challenges they face in new and creative ways.

We're also reaching out to senior leaders in the software community. One of the tremendous benefits of the conference in years past was to provide a platform for military leadership to provide guidance and perspective as well as share their vision of the challenges and opportunities facing the software community. SSTC2012 will revitalize that effort.

In today's extremely frugal environment, SSTC leadership is more committed than ever to ensuring a conference experience that returns each attendee back to their workplace more capable of do their job.

Upcoming Events

Visit <http://www.crosstalkonline.org/events> for an up-to-date list of events.

INCOSE International Workshop 2012
21-24 January 2012
Jacksonville, FL
<http://www.incose.org/newsevents/events/details.aspx?id=140>

National Security Technology Expo
6-8 February 2012
San Diego, CA
<http://www.ubm.com>

28th Annual National Test and Evaluation Conference
12-15 March 2012
Hilton Head Island, SC
<http://www.ndia.org/meetings/2910/Pages/default.aspx>

IEEE International Systems Conference
19-23 March 2012
Vancouver, BC
<http://ieeesyscon.org>

SEPG North America 2012
23 March 2012
Albuquerque, NM
<http://www.sei.cmu.edu/sepg/na/2012>

28th Annual National Logistics Conference
26-29 March 2012
Miami, FL
<http://www.ndia.org/meetings/2730/Pages/default.aspx>

Software Assurance Forum - Spring 2012
26-30 March 2012
McLean, VA
<https://buildsecurityin.us-cert.gov/bsi/events.html>

GovSec 2012
2-4 April 2012
Washington, DC
<http://govsecinfo.com/Home.aspx>

Systems and Software Technology Conference
23-26 April 2012
Salt Lake City, UT
<http://sstc-online.org>

SEPG Europe 2012
5-7 June 2012
Madrid, Spain
<http://www.sei.cmu.edu/sepg/europe/2012>

International Symposium 2012
9-12 July 2012
Roma, Italy
<http://www.incose.org/newsevents/events/details.aspx?id=142>

GFIRST8
19-24 August 2012
Atlanta, GA
<http://www.us-cert.gov/GFIRST>

26th International Biometrics Conference
26-28 August 2012
Kobe, Japan
<http://www.ourglocal.com/event/?eventid=11988>

15th Annual Systems Engineering Conference
22-25 October 2012
San Diego, CA
<http://www.ndia.org/meetings/3870/Pages/default.aspx>

Maturity
Or the Lack of It!

It is extremely odd that I was asked to write the BackTalk for this issue given that the theme has to do with maturity. For those who have never met me, you can rest assured that while I might grow older, I steadfastly refuse to grow up. Maturity? It must apply to my coding skills and certainly not to my behavior.

I have been a coder since the 1960s. I always thought I was pretty good. Luckily for me, back in 1997, I became an instructor for Personal Software Process (PSP), and later also qualified as an instructor for the Team Software Process (TSP). My coding skills truly matured. And, while I never taught any CMM® classes, I certainly knew the value of the CMM. It allowed me to assess the maturity of my organization, just as the PSP and TSP allowed me to assess the maturity of myself and my team, respectively.

But, how do I assess the maturity of my code itself? Is it the formatting? The documentation? The self-evident identifier names I use? Is there a way to look at a piece of code, and say, "Yes, this is very mature code?" Well if it is COBOL, that certainly makes it mature, right? And—if it IS mature code—do other developers appreciate the value and worth of the code? As a matter of fact, what about code that is not just "mature" in terms of high quality, but "mature" in terms of age?

One of my students recently pointed me to an excellent article by Joel Spolsky regarding code maturity. In it he discusses the value of "old code" and makes the observation that newer programmers always want to throw away old code figuring that it is bad. They are most often always wrong! He points out that new programmers treat old code as if it somehow rusted; the mere fact of it getting old makes it somehow less correct or useful. As Joel accurately points out, there is absolutely no reason to suspect that you are going to do a better job than whoever wrote the code "back in the day." And, of course, as all of us mature coders know, old code represents an accumulation of knowledge that took potentially years and years to accumulate. Mature code has been refined, sifted through the sieve of testing, and then strengthened by multiple updates and improvements. The mature code, if thrown away, will require yet another generation of programmers to find out that the old code is complex and hard to read only because the actual problem it solves is equally complex and hard to understand.

Having pointed this out, how do you know if you are one of "those coders" who write old code? Well, having the Internet as a guide, I have compiled a short list of items that might help tell you if you are one of those developers who write "old code." Do any of these apply to you?

- 6 a.m. is when you get up, not when you go to bed.
- You hear your favorite song on an elevator.
- You watch the Weather Channel. And you carry an umbrella.
- Jeans and a sweater no longer qualify as "dressed up."
- You do not know what time Taco Bell closes anymore.
- Your car insurance goes down instead of up.
- You feed your dog Science Diet instead of McDonalds leftovers.
- Sleeping on the couch makes your back hurt. And, if you do fall asleep on the couch, it takes two tries to get up.
- You no longer take naps from noon to 6 p.m. You want to, but you just can not find the time.
- Movies are not loud enough.
- Friends that call after 9 p.m. start the conversation out with "Did I wake you?"
- You know your pharmacists name.
- You start getting the "senior discount" without asking for it.
- Dinner and a movie is the whole date instead of the beginning of one.
- You actually eat breakfast food at breakfast time. Anytime you buy cereal, it's for fiber content, not the cute animal that advertises it.
- 90% or more of the time you spend in front of a computer is for real work.
- Grocery lists are longer than macaroni & cheese, diet Pepsi and Ho-Hos.
- You have read this list, and realized that several items apply to you. And it got less and less funny as you kept reading.

Have no fear, my friends. We are all getting "mature."

So is your code. And it is valuable. Resist the urge to simply rewrite something because it is not new. Not everything you compile has to be in Ruby, C#, or Javascript.

Cars grow old, and might become valuable antiques. Mostly, however, they simply wear out and are replaced. Luckily, your code is not an automobile. If it survived the first few years of use it only becomes more and more valuable. It has been debugged. Not only does it work, it represents accumulated knowledge. It works so well, you probably cannot afford to get rid of it.

Much like us mature developers.

David A. Cook, Ph.D.
Stephen F. Austin State University

CMM® is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

CrossTalk thanks the above organizations for providing their support.